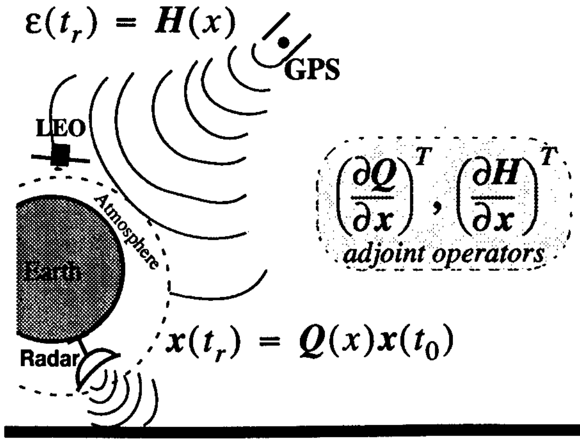
July 1998

NCAR TN-437+IA

A User's Guide to the MM5 Adjoint Modeling System

X. Zou, W. Huang and Q. Xiao


$\varepsilon(t_r) = H(x)$
GPS
LEO
Atmosphere
Earth
Radar
$\left(\frac{\partial Q}{\partial x}\right)^T, \left(\frac{\partial H}{\partial x}\right)^T$
adjoint operators
$x(t_r) = Q(x)x(t_0)$

Mesoscale and Microscale Meteorology Division
National Center for Atmospheric Research

CONTENTS

Acknowledgments

This user's guide is our first attempt to compile our adjoint code into written documentation, and it was not completely ready in time for our first MM5 Adjoint Model Tutorial which was held during July 24-25, 1997. This version is completed at the time of the second MM5 Adjoint Tutorial being held July 28-31, 1998.

We would like to take this opportunity to thank the reviewers (Juanzhen Sun and F. Vandenberghe) for their suggestions, which have added to the document's readability as well as technical accuracy, and for their willingness to complete their reviews within the minimum time they were allowed. We would also like to specifically thank Ingrid Moore who did all the editorial work for us.

INTRODUCTION

1

1 INTRODUCTION

During three and one-half year period from 1994 to 1997 when the first author stayed at NCAR, an adjoint mesoscale model suitable for analyzing and predicting mesometeorological flows was developed by the Mesoscale Prediction Group (MPG) within the Mesoscale and Microscale Meteorology (MMM) Division of the National Center for Atmospheric Research (NCAR). It is based on the fifth-generation Penn State/NCAR Mesoscale Model (MM5, Grell et al., 1993, Dudhia, 1993). MM5 is the latest in a series that developed from a mesoscale model described by Anthes and Warner (1978) with the addition of a multiple-nest capability, nonhydrostatic dynamics, and more physics options.

The MM5 modeling system is maintained by the MesoUser manager at NCAR. We hope that in the near future we will have a MesoUser manager for the MM5 adjoint modeling system, who will serve a similar role as the MesoUser manager of the MM5 modeling system, i. e., 1) maintaining a working set of programs and the accompanying set of C-shell scripts; 2) addressing and notifying users of uncovered problems; 3) testing bug fixes to the standard package prior to general release; and 4) providing user services. Right now these duties are shared by the members in the data assimilation subgroup within the MPG.

The current version of the MM5 adjoint modeling system and its prototype 4-dimensional data assimilation procedure were built on an original MM5 modeling system and executed before the forward model integration and after all the pre-processings of the data were done. Figure 1.1 shows the general flow of the set of programs required for a typical data assimilation and the subsequent forecast run. The rectangular boxes in the right column of the figure represent the programs that are run before an actual forward model prediction is made, and from top to bottom they represent the order in which the job decks are typically run. These are the same as in the MM5 modeling system. The middle part of the figure represents all of the non-conventional data, the "4D-VAR" experiments, or the sensitivity study. The forward model integration following either the INTERP or adjoint experiments is basically the same as in the original MM5 modeling system with minor changes. The last step of post-processing is the same as that in the MM5 modeling system. As we can see from this flow chart, we don't have a complete 4D-VAR system, which shall replace all the pre-processing programs in a consistent manner.

This document focuses primarily on explaining the use of the individual programs which employ the MM5 adjoint model. We will explain the job deck, parameter setting, and the technical aspects of the code of the main programs. We will not attempt to explain the technical aspects of other subroutines, which can be used as a black box for users anyway. There is a chapter for each program of the adjoint system. Each of the chapters reviews the user input required for the successful run and provides hints gleaned from experience. Examples with required input data are

MM5 "4D-VAR" or Adjoint Sensitivity Experiment

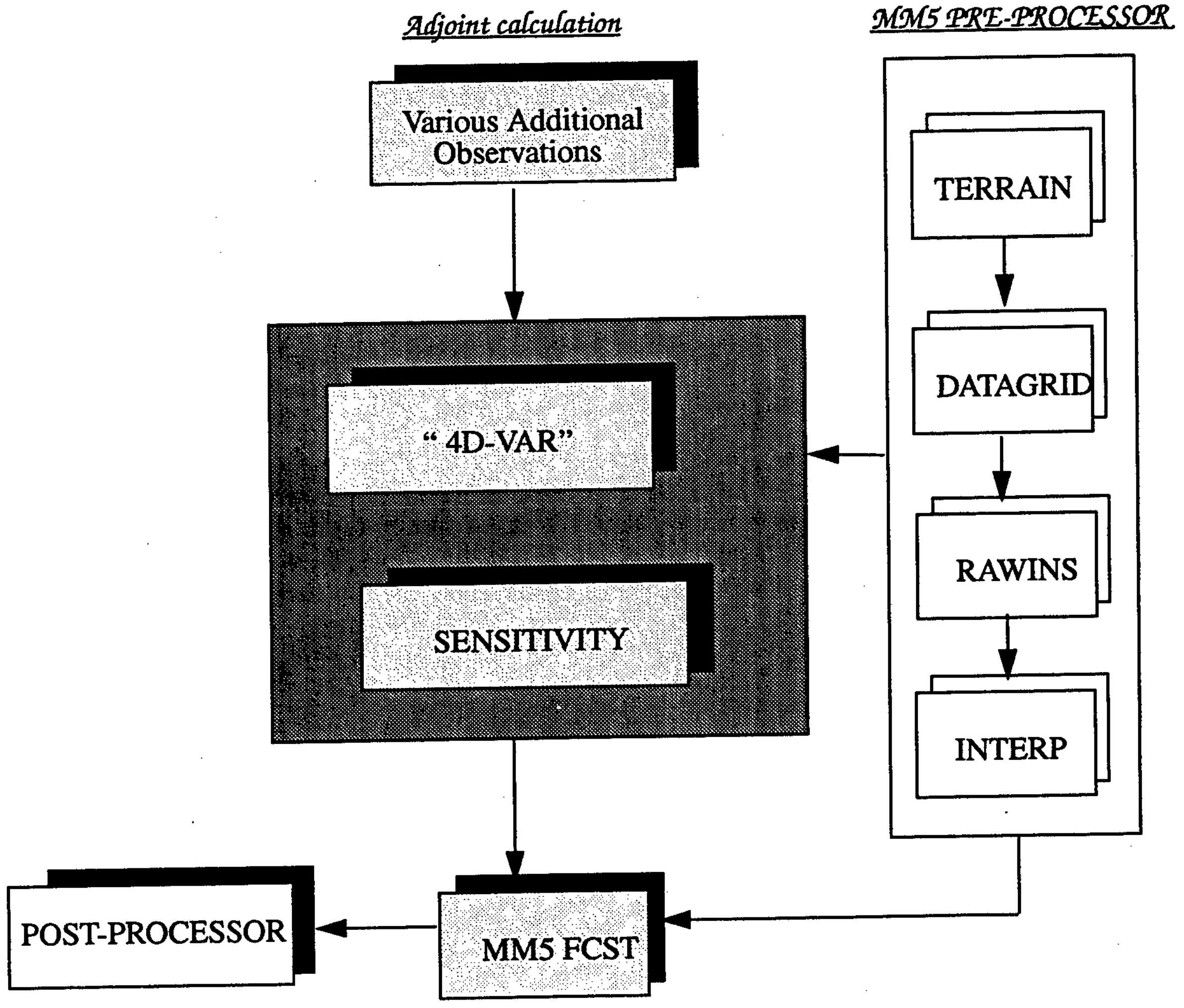


Fig. 1.1 Program flow for incorporating MM5 adjoint calculations: center rectangles denote individual components of the modeling and assimilation system. Shaded rectangles represent added programs to the MM5 modeling system with the use of the MM5 adjoint model.

provided and the expected results are displayed.

The remainder of this chapter addresses some of the fundamental aspects of the MM5 modeling system, which will potentially affect the construction and the use of the MM5 adjoint model in different ways. These are the vertical and horizontal grids, the time integration scheme, and the handling of the lateral boundary conditions. Following this is chapter 2, which introduces the user to the code structure particular to the MM5 adjoint modeling system. Chapter 3 briefly describes MM5 forward model run. Chapters 4 and 5 describe the sample jobs of two sensitivity experiments and a "4D-VAR" twin experiment, respectively. Both of them are available to the user. Conclusions and future planning are presented in Chapter 6.

1.1 Horizontal Grid Specification

The MM5 adjoint modeling system does not have a nesting feature. The horizontal domain is the Arakawa B-Grid, where the horizontal wind components (u, and v) are defined on the "dot" points and the temperature, moisture, vertical velocity and pressure perturbation fields (T, qv qr, qc, w, and p') on the "cross" points. The dimension of the model grids must be set up to allow the grid to begin and end on a dot point, which means that the total number of the grid points in both I and J directions are odd numbers. As in the MM5, model domain is a rectangular array of grid points, with J-direction (x-direction) moving from the left edge to the right edge and I-direct (y-direction) from the lower edge to the upper edge.

1.2 Vertical coordinate

The model uses σ, a terrain-following normalized pressure coordinate defined as

$$\sigma = (p - p_{top})/(p_{sfc} - p_{top}) \qquad (1.1)$$

where p_{top} is the user defined model lid (which normally is greater than 50 mb) and p_{sfc} is the surface pressure. The model computations are performed on the half σ layers, where the values of u,v,T,qv,qc,qr,w,and p' are taken to be representative of the respective variables through the depth of the σ layer.

1.3 Time integration

The nonhydrostatic equations of MM5 are fully compressible and they permit sound waves. These are fast waves and require a short time step for numerical stability. A time-splitting scheme is applied to split these fast moving waves from the rest of the solution. Terms related to the sound waves are separated from the slow terms, which are kept constant during the sub-steps, and a short time step is used to make the model more efficient. A semi-implicit scheme is used for the fast terms and a leapfrog scheme is used for the slow terms.

1.4 Lateral boundary condition

The lateral boundary conditions are brought to the model through a nudging process, except for the vertical velocity and the horizontal wind at the outflow boundaries. The vertical velocity can vary freely except for the outmost rows and columns to satisfy a zero gradient condition. The wind values at the outflow boundaries are obtained by extrapolation from the interior points.

GETTING STARTED

2

2 Getting Started

This section is for new users of the MM5 adjoint model. It is assumed that the user is familiar with the MM5 modeling system and with the UNICOS environment.

Section 2.1 of this chapter describes the files that are currently available in the supported program directories. A brief overview of the standard naming convention of these files is given. Section 2.2 describe a supported test capability to check the correctness of the TLM and adjoint model. The last portion of the chapter, section 2.3, describes how users can access the Fortran source code for the MM5 adjoint model system.

2.1 File Naming Conventions

All of the adjoint-related calculations are controlled by job decks. The individual user-level modeling system programs are FORTRAN source codes that are maintained by MAKE (a Cray source code control system). MAKE allows users to save tremendous compiling time for any modification made to the code.

The job decks and files for each component of the package are located in separate subdirectories on paiute. Following is a list of the main subdirectories containing all the MM5 adjoint modeling system files.

- **fdvar**

  ~fdvar/Tutorial_97/adjm
  ~fdvar/Tutorial_97/fwdm
  ~fdvar/Tutorial_97/include
  ~fdvar/Tutorial_97/main_fcst
  ~fdvar/Tutorial_97/main_minimization
  ~fdvar/Tutorial_97/main_sensitivity
  ~fdvar/Tutorial_97/main_test
  ~fdvar/Tutorial_97/mfwm
  ~fdvar/Tutorial_97/minm
  ~fdvar/Tutorial_97/tglm
  ~fdvar/Tutorial_97/tstm
  ~fdvar/Tutorial_97/utlm

To run a standard model forecast, sensitivity experiment, or data assimilation experiment, users are required to make a few lines of changes in the *include* subdirectory and either *main_fcst* (model forecast), or *main_sensitivity* (adjoint sensitivity calculation), or *main_minimization* (4D-VAR experiment) subdirectory, respectively. Users do not need to touch the rest of the subdirectories, i.e., *adjm, bdym, fwdm, mfwdm, minm, tglm, utlm, and main_test.* In this section, we will briefly describe these directories which users do not need to modify. The subdirectory *main_test* will be explained in this section. The three main subdirectories *main_fcst* , *main_sensitivity* and *main_minimization* will be explained in more detail in the following chapters 3-5.

The subdirectory *fwdm* contains all the subroutines of the MM5 model which are listed as follows:

- **fwdm**

```
~fdvar/Tutorial_97/fwdm/addall.F
~fdvar/Tutorial_97/fwdm/addrx1c.F
~fdvar/Tutorial_97/fwdm/bdy_store.F
~fdvar/Tutorial_97/fwdm/bdy_update.F
~fdvar/Tutorial_97/fwdm/bdyin.F
~fdvar/Tutorial_97/fwdm/bdyuv.F
~fdvar/Tutorial_97/fwdm/bdyval.F
~fdvar/Tutorial_97/fwdm/blkpbl.F
~fdvar/Tutorial_97/fwdm/cadjmx.F
~fdvar/Tutorial_97/fwdm/cup.F
~fdvar/Tutorial_97/fwdm/cupara1.F
~fdvar/Tutorial_97/fwdm/cupara2.F
~fdvar/Tutorial_97/fwdm/dots.F
~fdvar/Tutorial_97/fwdm/equate.F
~fdvar/Tutorial_97/fwdm/exmoiss.F
~fdvar/Tutorial_97/fwdm/gauss.F
~fdvar/Tutorial_97/fwdm/hadv.F
~fdvar/Tutorial_97/fwdm/hirpbl.F
~fdvar/Tutorial_97/fwdm/init.F
~fdvar/Tutorial_97/fwdm/lwrad.F
~fdvar/Tutorial_97/fwdm/mapsmp.F
~fdvar/Tutorial_97/fwdm/maximi.F
~fdvar/Tutorial_97/fwdm/minimi.F
~fdvar/Tutorial_97/fwdm/minit.F
~fdvar/Tutorial_97/fwdm/minit0.F
~fdvar/Tutorial_97/fwdm/mrdinit.F
~fdvar/Tutorial_97/fwdm/nlmod.F
~fdvar/Tutorial_97/fwdm/nudge.F
~fdvar/Tutorial_97/fwdm/outprt.F
~fdvar/Tutorial_97/fwdm/output.F
~fdvar/Tutorial_97/fwdm/outsav.F
~fdvar/Tutorial_97/fwdm/outtap.F
~fdvar/Tutorial_97/fwdm/p1p2.F
~fdvar/Tutorial_97/fwdm/param.F
~fdvar/Tutorial_97/fwdm/pwater.F
~fdvar/Tutorial_97/fwdm/conadv.F
```

~fdvar/Tutorial_97/fwdm/conmas.F
~fdvar/Tutorial_97/fwdm/convad.F
~fdvar/Tutorial_97/fwdm/decpu.F
~fdvar/Tutorial_97/fwdm/deltatim.F
~fdvar/Tutorial_97/fwdm/diffu.F
~fdvar/Tutorial_97/fwdm/diffut.F
~fdvar/Tutorial_97/fwdm/rdinit.F
~fdvar/Tutorial_97/fwdm/set_assign.F
~fdvar/Tutorial_97/fwdm/sfcrad.F
~fdvar/Tutorial_97/fwdm/skipf.F
~fdvar/Tutorial_97/fwdm/slab.F
~fdvar/Tutorial_97/fwdm/solar1.F
~fdvar/Tutorial_97/fwdm/solve3.F
~fdvar/Tutorial_97/fwdm/sound.F
~fdvar/Tutorial_97/fwdm/swrad.F
~fdvar/Tutorial_97/fwdm/tmass.F
~fdvar/Tutorial_97/fwdm/ua2ueb.F
~fdvar/Tutorial_97/fwdm/vadv.F
~fdvar/Tutorial_97/fwdm/vtran.F
~fdvar/Tutorial_97/fwdm/wbc.F
~fdvar/Tutorial_97/fwdm/xa2xb.F

We can see that most of the subroutines are copied from the MM5 modeling system except the subroutines *ua2ueb.F, xa2xb.F, bdy_update.F* and *wbc.F.* All of these are needed from the fact that for a model to be used in an optimization procedure, only the independent input variables should be viewed as the control variables of the model. (see Zou et al., 1997). The subroutine *ua2ueb.F* puts the initial condition (IC) near the lateral boundary to its corresponding boundary values; *xa2xb.F* gives the UA, VA, ... values to UB, VB, ...; *bdy_update.F* calculates values of the lateral boundary conditions (LBC) for UEB, UWB, ..., etc at the next LBC time from its previous values of UEB, UWB, ..., and their boundary tendencies; and *wbc.F* calculates the vertical velocity at both the bottom and top of the model from other variables, and is part of the calculation originally done in the pre-process stage in the MM5 modeling system.

The subdirectory *tglm* includes all the linearized subroutines corresponding to each of the nonlinear subroutines in the subdirectory *fwdm.* The linearized files have a standard naming convention. The first letter of the subroutine is always "L", followed by the same name as in the MM5 forward model (those files in the subdirectory *fwdm*).

- **tglm**

~fdvar/Tutorial_97/tglm/lbdyval.F
~fdvar/Tutorial_97/tglm/lblkpbl.F
~fdvar/Tutorial_97/tglm/lcadjmx.F
~fdvar/Tutorial_97/tglm/lconvad.F
~fdvar/Tutorial_97/tglm/ldecpu.F
~fdvar/Tutorial_97/tglm/ldiffu.F
~fdvar/Tutorial_97/tglm/ldiffut.F
~fdvar/Tutorial_97/tglm/lgauss.F
~fdvar/Tutorial_97/tglm/lhadv.F

```
~fdvar/Tutorial_97/tglm/lsfcrad.F
~fdvar/Tutorial_97/tglm/lslab.F
~fdvar/Tutorial_97/tglm/lsolve3.F
~fdvar/Tutorial_97/tglm/lsound.F
~fdvar/Tutorial_97/tglm/lvadv.F
```

The total number of files in *tglm* is much less than that in *fwdm*, since the same subroutines, which are linear, will be used in the TLM. To make this happen, we have kept the same variable notation for perturbation fields in TLM as in the nonlinear model, and the basic state variables are appended by a number "9".

All the adjoint subroutines are kept in the subdirectory *adjm*. For each subroutine in MM5, there is an adjoint subroutine with the first letter being "A" followed by the same name as in the MM5. For instance, the adjoint subroutine ahadv.F is simply the transpose of the linearized subroutine lhadv.F of hadv.F, realized at the coding level. The adjoint variables have the same naming convention in the code as their corresponding variables in the MM5 or their perturbation variables in the TLM. Following is a list of the subdirectory *adjm*:

- **adjm**

```
~fdvar/Tutorial_97/adjm/abdy_update.F
~fdvar/Tutorial_97/adjm/abdyuv.F
~fdvar/Tutorial_97/adjm/abdyval.F
~fdvar/Tutorial_97/adjm/ablkpbl.F
~fdvar/Tutorial_97/adjm/acadjmx.F
~fdvar/Tutorial_97/adjm/aconvad.F
~fdvar/Tutorial_97/adjm/adecpu.F
~fdvar/Tutorial_97/adjm/adiffu.F
~fdvar/Tutorial_97/adjm/adiffut.F
~fdvar/Tutorial_97/adjm/aequate.F
~fdvar/Tutorial_97/adjm/agauss.F
~fdvar/Tutorial_97/adjm/ahadv.F
~fdvar/Tutorial_97/adjm/anudge.F
~fdvar/Tutorial_97/adjm/apwater.F
~fdvar/Tutorial_97/adjm/asfcrad.F
~fdvar/Tutorial_97/adjm/aslab.F
~fdvar/Tutorial_97/adjm/asolar1.F
~fdvar/Tutorial_97/adjm/asolve3.F
~fdvar/Tutorial_97/adjm/asound.F
~fdvar/Tutorial_97/adjm/avadv.F
~fdvar/Tutorial_97/adjm/awbc.F
```

Both TLM and the adjoint model are linear models, which are all linearized around the nonlinear model solution. The basic principles we adapted in MM5 TLM and adjoint model development for calculating the nonlinear coefficients are as follows:

The model predictions at every time step, consisting of wind, temperature, specific humidity, cloudwater, rainwater, and pressure perturbation, were saved and used as the only input basic state information. The nonlinear coefficients in the TLM and adjoint model are all recalculated from them.

Also, for every subroutine, we assume that the input variables and their basic state values are available and consist of all the input data for that subroutine. This way each of the subroutines becomes a relatively independent piece of code in developing and checking its tangent linear and adjoint code and linking them together. Since all the basic state variables are appended by "9", there is a need to generate another set of subroutines for the basic state calculation in the TLM and adjoint model if these subroutines in the nonlinear model produced some values which were used in the future calculation of the nonlinear coefficients in the TLM and adjoint model. These subroutines are assembled in the subdirectories *mfwm,* which is listed as follows:

- **mfwm**

  ~fdvar/Tutorial_97/mfwm/mbdyuv.F
  ~fdvar/Tutorial_97/mfwm/mbdyval.F
  ~fdvar/Tutorial_97/mfwm/mblkpbl.F
  ~fdvar/Tutorial_97/mfwm/mconadv.F
  ~fdvar/Tutorial_97/mfwm/mconmas.F
  ~fdvar/Tutorial_97/mfwm/mconvad.F
  ~fdvar/Tutorial_97/mfwm/mdecpu.F
  ~fdvar/Tutorial_97/mfwm/msfcrad.F
  ~fdvar/Tutorial_97/mfwm/mslab.F
  ~fdvar/Tutorial_97/mfwm/msound.F

Notice that all the subroutines in *mfwm* subdirectory start with a letter "M", followed by the same subroutine name in the MM5. For instance, subroutine *mblkpbl.F* is the same as *blkpbl.F* in the subdirectory *fwdm* except that the input and output variables of *blkpbl.F* are appended with "9" in *mblkpbl.F.* The local variables in *mblkpbl.F* are still the same as in *blkpbl.F*

Having all the codes for the MM5, its TLM and its adjoint model, we now move to the code for the minimization procedure. The subdirectory *minm* contains all the subroutines for the limited-memory quasi-Newton method (Zou et al., 1993). Following is a list of them:

- **minm**

  ~fdvar/Tutorial_97/minm/ddot.F
  ~fdvar/Tutorial_97/minm/lmcheck.F
  ~fdvar/Tutorial_97/minm/lmdir.F
  ~fdvar/Tutorial_97/minm/lmprint.F
  ~fdvar/Tutorial_97/minm/lmstep.F
  ~fdvar/Tutorial_97/minm/va15ad.F
  ~fdvar/Tutorial_97/minm/va15bd.F
  ~fdvar/Tutorial_97/minm/va15cd.F
  ~fdvar/Tutorial_97/minm/vd05ad.F
  ~fdvar/Tutorial_97/minm/vd05bd.F
  ~fdvar/Tutorial_97/minm/verigr.F

In the minimization code, one dimensional arrays are used for the contral variable, gradient, and so on, it is necessary to convert the multi-dimensional variables in the model into one dimensional variable. The subroutines for data transfer from multi-variables and multi-dimensions to one dimensional array and back, along the weighting and scaling, the direct access of data, the interactive mass storage usage and so on, are included in the subdirectory *utlm.*

- **utlm**

~fdvar/Tutorial_97/utlm/add_variable.F
~fdvar/Tutorial_97/utlm/aget.F
~fdvar/Tutorial_97/utlm/aget0.F
~fdvar/Tutorial_97/utlm/aput.F
~fdvar/Tutorial_97/utlm/aput0.F
~fdvar/Tutorial_97/utlm/aremove_negative.F
~fdvar/Tutorial_97/utlm/astart.F
~fdvar/Tutorial_97/utlm/averg.F
~fdvar/Tutorial_97/utlm/bas_appro.F
~fdvar/Tutorial_97/utlm/bas_approw.F
~fdvar/Tutorial_97/utlm/diff_variable.F
~fdvar/Tutorial_97/utlm/maxdiff.F
~fdvar/Tutorial_97/utlm/maxdiffb.F
~fdvar/Tutorial_97/utlm/oned.F
~fdvar/Tutorial_97/utlm/p1p2.F
~fdvar/Tutorial_97/utlm/prnwgt.F
~fdvar/Tutorial_97/utlm/rcvr_rst.F
~fdvar/Tutorial_97/utlm/remove_negative.F
~fdvar/Tutorial_97/utlm/save_rst.F
~fdvar/Tutorial_97/utlm/scaling.F
~fdvar/Tutorial_97/utlm/scalingb.F
~fdvar/Tutorial_97/utlm/trans_bas.F
~fdvar/Tutorial_97/utlm/transf.F
~fdvar/Tutorial_97/utlm/transfb.F
~fdvar/Tutorial_97/utlm/transfb_grad.F
~fdvar/Tutorial_97/utlm/transhir.F
~fdvar/Tutorial_97/utlm/weight.F
~fdvar/Tutorial_97/utlm/write_mss.F
~fdvar/Tutorial_97/utlm/zero_variable.F

All the code listed above does not need to be modified for different case studies and different adjoint applications.

2.2 Code Testing

The correctness of the MM5 TLM and adjoint model can be tested by a Taylor formula and an algebraic expression, respectively (see (3.15) in Zou et al. 1997). We provided this testing code in the subdirectory *main_adjoint_test.*

The job deck *tgl.deck* will carry out a TLM test. For an arbitrarily selected predicted variable, the value of its perturbation prediction obtained by TLM should approach the NLM predicted values linearly as the order of magnitude of the initial perturbation decreases. This will show a list of results in the output file as follows:

$\phi_u(\alpha)$	$\phi_v(\alpha)$	$\phi_T(\alpha)$	$\phi_q(\alpha)$
0.1172015E+1	0.1295176E+1	0.1845927E+1	0.5209394E+1
0.6802895E+1	0.5621636E+1	0.3591716E+2	0.1351556E+3
0.1146562E+2	0.1018204E+2	0.8753406E+2	0.8990221E+3
0.1000003E+1	0.1000012E+1	0.9999943E+0	0.9999410E+0
0.1000002E+1	0.1000004E+1	0.9999994E+0	0.9999952E+0
0.1000001E+1	0.1000003E+1	0.9999997E+0	0.1000000E+1
0.1000001E+1	0.1000003E+1	0.9999997E+0	0.1000000E+1
0.1000001E+1	0.1000003E+1	0.9999993E+0	0.1000000E+1
0.1000002E+1	0.1000002E+1	0.9999973E+0	0.9999975E+0
0.9999699E+0	0.9999728E+0	0.9999809E+0	0.1000045E+1
0.9999190E+0	0.9999589E+0	0.9998853E+0	0.9996812E+0
0.1001545E+1	0.1000918E+1	0.1004275E+1	0.1002713E+1
0.1053077E+1	0.1021803E+1	0.1207661E+1	0.1111543E+1
0.8635745E+1	0.6896338E+1	0.2222289E+2	0.6861195E+1

The above results were obtained by running the standard case for a 3-h prediction of zonal wind component, meridional wind component, temperature and specific humidity. The standard case is the same case chosen for the MM5 tutorial since 1997. It is set up to make a 24-h forecast for the time period 0000 UTC 13 to 0000 UTC 14 March 1993 at a very coarse resolution (120 km) with a grid size of 26 x 31 x 10. The values of α varies from 10^{-3} to 10^{-16}. $\boldsymbol{h}$ is taken as the vector of the initial condition. Physical options include the grid-resolvable precipitation, the Kuo cumulus precipitation, the bulk PBL and surface flux. All the test and experimental results in this user guide are obtained using the same version of the MM5 forward model, the corresponding tangent linear model and adjoint model.

The job deck *adj.deck* will run the adjoint test. A correct adjoint code means that the two numbers (Wright and Wleft represent the values of the right- and left-hand-side of the algebraic formula (3.16) in Zou et al. (1997)) are the same with 13 digit accuracy on a 64-bit machine. Exception when less digits could be obtained even with an actually correct code may occur, which was discussed in Zou et al. (1997). An example of the correctness check of the adjoint model is shown below:

```
Wright  =  0.135194342l0906E+17
Wleft   =  0.13519434210906E+17
```

which was obtained by running the standard case for a 3-h time integration. It includes the 3-h prediction of all the model fields. Since no normalization was done, such a test may be misleading for the fact that some of the quantities (say T or p') may be several orders of magnitude bigger than others (say u, v) and a correct test may just mean that the largest term is correct. A more strict test is to test different variables separately. For example, we can test only the wind fields (u, v) and we obtain:

```
Wright  =  0.53709592507228E+13
Wleft   =  0.53709592507228E+13
```

or we can make a test to the pressure perturbation (p'):

```
Wright  =  0.13512767057137E+17
Wleft   =  0.13512767057137E+17
```

We observe that if there is a bug in the adjoint model which affects the wind fields and results in only a 10 digit accuracy, it won't be reflected in a test which includes all the model fields.

Users are advised to make their own TLM and adjoint tests for their new cases before making a productive sensitivity or data assimilation experiment. For the latter, a gradient check is also necessary, which will be discussed in Chapter 5.

2.3 Adjoint Modeling System Source Code

The standard version of the MM5 adjoint modeling system can be copied from the NCAR cray paiute or read from the mass storage file. They are listed as

paiute: ~fdvar/Tutorial_97/MM5_ADJ.tar.gz

or

mass storage: /FDVAR/TUTORIAL/MM5_ADJ.tar.gz

Users just need to execute two commands: gunzip MM5_ADJ.tar.gz and tar -xvf MM5_ADJ.tar to get the standard source code after the files are copied to their local machine.

Remark:

A detailed description of the Make utility and the job deck syntax is given in Chapters 2 and 3 of the "PSU/NCAR Mesoscale Modeling System Tutorial Class Notes.".

MM5FCST

3

3 MM5 Forecast

3.1 Purpose

This chapter describes how to do a straight-forward model integration. Users have a choice to use the standard MM5 modeling system to run the forecast or to use the code consistent with the MM5 adjoint model. If it's the latter case, users can carry out the forward MM5 model forecast in the subdirectory *main_fcst*, which contains only two files:

- **main_fcst**

```
~fdvar/Tutorial_97/main_fcst/fcst.deck
~fdvar/Tutorial_97/main_fcst/mm5fcst.F
```

3.2 Forecast Job deck

The job deck *fcst.deck* is a C-shell script and executes a forward model run using codes (*mm5fcst.F*) consistent with the MM5 adjoint model. It is set up to make a 24-h forecast starting either from the standard MM5 input initial condition or the "optimal" initial condition obtained by a minimization procedure. The user may define shell variables that are expanded for use as file names, directories, or titles for output files. Users with syntax questions for C-shell constructs are referred to the man pages for "csh".

The job deck is structured similarly to the job decks in the MM5 modeling system, i.e., one can choose different physics options, set the IC and LBC, and output the forecasts at the desired time interval. There are only two parameters in the job deck which may need further explanation: parameter IFWDRUN and LRG_PRE. The parameter IFWDRUN=0 means to make a forecast from the standard MM5 IC file, and IFWDRUN=1 means that one wants to make a forecast from an optimal IC (output from a 4D-VAR run) which is formatted as

```
OPEN (UNIT=91,FORM='UNFORMATTED',STATUS='UNKNOWN')
READ (91) UA,VA,TA,QVA,PPA,WA,QCA,QRA
CLOSE(91)
```

The parameter LRG_PRE controls whether or not to include the large-scale precipitation process. We added this parameter in case one wants to examine the moisture "on-off" problem. In the following we enclose the job deck *fcst.deck.*

- **fcst.deck**

```
# QSUB -r FCST                # request name
# QSUB -q reg                # job queue class prem  econ
# QSUB -eo                  #
# QSUB -o  fcst.out           # stdout and stderr together
# QSUB -lM 8Mw                 # maximum memory
# QSUB -lT 5000               # time limit
# QSUB                     # no more qsub commands
#
 ja
#
 set echo
#          *******************************************
#          *******     4dvar batch C shell     *******
#          *******************************************
# set Data_Dir = /Models/zou/TUTORIAL/FDVAR/data
 set Work_Dir = /Models/zou/tmp-fcst
 set Curt_Dir = `pwd`
 cd $Work_Dir

#set Recompiled = Yes       # recompile the code
 set Recompiled = No
#
#     type of 4dvar job
#
 set IFWDRUN = 1            # 0 -> From analysis, 1 -> From Optimal IC
#
#     local namelist values
#
cat >! mmlif << EOF
&FDPARAM
 IFWDRUN  = $IFWDRUN,     ; 0 -> From analysis, 1 -> From Optial IC
LRG_PRE  = 1,     ; 0,1 large-scale precipitation
 & ;-----------------------------------------------------------------
 &OPARAM                                ;        <-- MOD2
 ;-----------------------------------------------------------------
 ;     YOU CAN REMOVE THE UNWANTED DATA FROM THE FOLLOWING LISTING
 ;     AND USE THE DEFAULT VALUES DEFINED IN SUBROUTINE 'PARAM'.
 ;-----------------------------------------------------------------
 IFREST = F,     ;RESTART
   IXTIMR  =720,
 LEVIDN  = 0, ; level of nest for each domain
 NUMNC   = 1, ; ID of mother domain for each nest
 IFSAVE = F,      ; SAVE DATA FOR RESTART
   SAVFRQ =180., ;  ... in minutes.
 IFTAPE = 1,      ; OUTPUT FOR GRIN backend
   TAPFRQ =60., ;  ... in minutes.
 IFPRT = 1,       ; do not change
```

```
PRTFRQ = 180.,   ; Print output frequency in minutes
MASCHK = 30,     ; MASS CONSERVATION CHECK (minutes)
& ;-----------------------------------------------------------------
&LPARAM
iactiv=1,0,0,0,0,0,0,0,0,0, ; in case of restart: was this domain active?
;
;************* start physics options ****************
;
IFRAD    = 0,   ;RADIATION COOLING OF ATMOSPHERE - 0, 1, 2
RADFRQ   = 30., :RADIATION FREQUENCY IN MINUTES
ICUSTB   = 1,   ;STABILITY CHECK FOR CUMULUS PARAM. - 0(no stab. check), 1
IEXICE   = 0,   ;ICE-PHYSICS IN EXPLICIT SCHEME - 0, 1
IFDRY    = 0,   ;FAKE-DRY RUN - 0, 1
IMVDIF   = 0,   ;MOIST VERTICAL DIFFUSION IN CLOUDS - 0, 1
IBMOIST  = 0,   ;BOUNDARY AND INITIAL WATER/ICE SPECIFIED - 0, 1
ICOR3D   = 1,   ;3D CORIOLIS FORCE - 0, 1
IFUPR    = 0,   ;UPPER RADIATIVE BOUNDARY CONDITION - 0, 1
;
IBOUDY = 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, ;BOUNDARY CONDITIONS - 0, 1, 2, 3, 4
IBLTYP = 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, ;PBL TYPE - 0, 1, 2
IDRY   = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;MOIST OR DRY CASE - 0, 1
IMOIST = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;NON-EXPLICIT, EXPLICIT, - 1, 2
ICUPA  = 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;NONE,KUO,GRELL,AS - 1,2,3,4
ISFFLX = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;SURFACE FLUXES - 0, 1
ITGFLG = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;SURFACE TEMPERATURE - 1, 3
ISFPAR = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;SURFACE CHARACTERISTICS - 0, 1
ICLOUD = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;CLOUD EFFECTS ON RADIATION - 0, 1
ICDCON = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;CONSTANT DRAG COEFFICIENTS - 0, 1
IFSNOW = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;SNOW COVER EFFECTS - 0, 1
IMOIAV = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;VARIABLE MOISTURE AVAILABILITY - 0, 1
IVMIXM = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;VERTICAL MIXING OF MOMENTUM - 0, 1
HYDPRE = 1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,;HYDRO EFFECTS OF LIQ WATER - 0., 1.
IEVAP  = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;EVAP OF CLOUD/RAINWATER - <0, 0, >0
ISHALLO= 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;SHALLOW CONVECTION - 0, 1
;
;************* end physics options ******************
;
;
;
;************* start nesting options ****************
;
nestix = 26,  46,  46,  46,  46,  46,  46,  46,  46,  46, ; domain size I
nestjx = 31,  61,  61,  61,  61,  61,  61,  61,  61,  61, ; domain size J
nesti  =  1,   1,   1,   1,   1,   1,   1,   1,   1,   1, ; start location I
nestj  =  1,   1,   1,   1,   1,   1,   1,   1,   1,   1, ; start location J
xstnes =  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,; domain initiation
xennes =1440.,720.,720.,720.,720.,720.,720.,720.,720.,720.,; domain completion
ioverw =  0,   0,   0,   0,   0,   0,   0,   0,   0,   0, ; overwrite domain
;
;************* end nesting options ******************
& ;-----------------------------------------------------------------
&PPARAM
TIMAX  = 1440.,  ; IN MINUTES. 720=12h, 1440=24h, 2160=36h, 2880=48h
ZZLND  = 0.1,    ; ROUGHNESS LENGTH OVER LAND IN METERS
```

```
ZZWTR  = 0.0001, ; ROUGHNESS LENGTH OVER WATER IN METERS
ALBLND = 0.15,  ; ALBEDO
THINLD = 0.04,  ; SURFACE THERMAL INERTIAL
XMAVA  = 0.3,   ; MOISTURE AVAILABILITY OVER LAND AS A DECIMAL
;               FRACTION OF ONE
CONF   = 1.0,   ; NON-CONVECTIVE PRECIPITATION SATURATION
TISTEP = 360.,  ; COARSE DOMAIN DT IN MODEL, USE 3*DX
ifeed  = 3,     ; OLD FEEDBACK, NO/LIGHT SMOOTHING IN FEEDBK - 1,2,3
iabsor = 0,     ; SPONGE ON UPPER BOUNDARY - 0,1
& ;-----------------------------------------------------------------
&FPARAM
& ;-----------------------------------------------------------------
EOF
#
#####################################################################
###################                    #######################
###################   END USER MODIFICATION   ######################
###################                    #######################
#####################################################################
#
#     initializations, no user modification required
#
set LETTERS = (A B C D E F G H I J K L M N O P Q R S T U V W X Y Z)
set FDvarUser = ~mesouser/MM5V1/MM5
#
 if ( $Recompiled == Yes ) then
#
  make
  mswrite -t 1000 mm5fcst.exe $OutMM/mm5fcst.exe
  ls -l
else
  msread mm5fcst.exe $OutMM/mm5fcst.exe
  chmod +x mm5fcst.exe
endif
#
#     set up fortran input files for the forecast
#
 rm fort.*
ln -s mmlif            fort.7
 ln -s $Data_Dir/bdyout                     fort.9
 ln -s $Data_Dir/guess_input_kl4dvar.o       fort.11
# ln -s $Data_Dir/guess_input_kl4dvar        fort.11
 ln -s $Data_Dir/DirObs_kl4dvar            fort.20
#
 execute:
#
#     run FDVAR
#
 date ln -s $Curt_Dir/mm5fcst.exe .
 timex nice mm5fcst.exe >&! $Curt_Dir/mm5fcst.print.out
 #
```

3.3 The Main forecast program: mm5fcst.F

About mm5fcst.F, we mention two points: (1) the variable MAXSTP isthe total time-steps for the entire forecast duration calculated from the value of TIMAX, whose value was set in the job deck; and the call to subroutine REMOVE_NEGATIVE, which is needed if the optimal IC is obtained without a specific constraint that the moisture variables qv should always be positive. In the following we enclose the main routine mm5fcst.F:

```
      PROGRAM MM5FCST
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C         Forward forecast from MM5 analysis or "optimal" IC
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCCCC
#     include <parame.incl>
#     include <parmin.incl>
#     include <param2.incl>
#     include <param3.incl>
#     include <addr0.incl>
#     include <various.incl>
#     include <point2d.incl>
#     include <dusolve1.incl>
#     include <variousn.incl>
#     include <point3d.incl>
#     include <pointbc.incl>
#     include <nonhyd.incl>
#     include <nonhydb.incl>
#     include <bcontrol.incl>
#     include <minim.incl>
#     include <obsdata.incl>
C
      DIMENSION AUA(MIX, MJX, MKX),AVA(MIX, MJX, MKX),
     -     ATA(MIX, MJX, MKX),AQVA(MIX, MJX, MKX),
     -     AWA(MIXNH,MJXNH,KXP1NH),APPA(MIXNH,MJXNH,MKXNH),
     -    ATGA(MIX,MJX), ARAINC(MIX,MJX), ARAINNC(MIX,MJX)
      DIMENSION AQCA(MIXM, MJXM ,MKXM ), AQRA(MIXM ,MJXM ,MKXM ),
     -     AQIA(MIXIC,MJXIC,MKXIC),AQNIA(MIXIC,MJXIC,MKXIC)
C
C*********************************************************************
      COMMON /IEXCMN/ IEXEC(MAXNES)
      DIMENSION IERR(2)
      COMMON /LRG_PRECIPITATION/LRG_PRE
      LRG_PRE=1
      READ (7,FDPARAM)
      PRINT FDPARAM
C
C--- INITIALIZE TO ZERO
C
      DO 1 N=1,MAXNES
      DO 1 I=1,IHUGE
         ALLARR(I,N)=0.
```

```
    1  CONTINUE
       DO 2 N=1,MAXNES
       DO 2 I=1,MIX
       DO 2 J=1,MJX
          XNUU(I,J,N)=0.
          XMUU(I,J,N)=0.
          XNUT(I,J,N)=0.
          XMUT(I,J,N)=0.
    2  CONTINUE
C
C--- FIRST THINGS FIRST: PRESET ADDRESS OF ALL VARIABLES
C--- FOR ALL NESTS.
C
       NSTTOT=1
       CALL ADDALL
C
C--- INITIALIZE NUMBER OF ACTIVE NESTS AND TOTAL POSSIBLE
C--- NESTS TO ZERO
C
       DO 4 N=1,NLNES
       DO 4 NN=1,MAXNES
          NUMLV(N,NN)=0
     4 CONTINUE
C
C--- FILL COARSE DOMAIN VARIABLES WITH DOMAIN (1)
C
       CALL ADDRX1C(IAXALL(1,1))
C
C-----SET UP PARAMETERS:
C
       DO 5 NN=1,MAXNES
          IEXEC(NN)=1
          LEVIDN(NN)=0
          NUMNC(NN)=0
     5 CONTINUE
       IF(MAXNES.EQ.1)THEN
           PRINT *,
      1 '****************** ONE DOMAIN ONLY!!! ***********'
       ELSE
           PRINT *,
      1 '****************** MULTI LEVEL RUN!!! ***********'
           PRINT *,
      1 '**************** ',MAXNES,' DOMAINS TOTAL *******'
       ENDIF
C
C--- READ IN PARAMETERS AND NAMELISTS
C
       CALL PARAM(IEXEC)
C  READ IN SUBSTRATE TEMPERATURE:
C
       READ(9) TMN
       CALL SKIPF (9,1,IERR)
C
C  READ IN THE BOUNDARY CONDITIONS:
```

```
C
             CALL  BDYIN( 9,TBDYBE ,BDYTIM ,IL,JL,IBMOIST)
C
C                  obtained values of  UEBC,UEBCT,......
C
C---- READ IN THE COARSE GRID INITIAL CONDITIONS
C
            CALL MRDINIT(11,IL,JL,0)
C
C                  obtained values of  UA,VA,.....WA
C
        CALL MINIT0(IEXEC,IL,JL)
C
C                  obtained values of  2-D fields
C
        IF (IFWDRUN .EQ. 1) THEN
           OPEN (UNIT=91,FORM='UNFORMATTED',STATUS='UNKNOWN')
           READ (91) UA,VA,TA,QVA,PPA,WA,QCA,QRA
           CLOSE(91)
           CALL REMOVE_NEGATIVE(QVA,QCA,QRA,QIA,QNIA,IEXMS,IICE,
     -           MIX,MJX,MKX,MIXM,MJXM,MKXM,MIXIC,MJXIC,MKXIC,
     -           IQINDX,IQCINDX,IQRINDX,IQIINDX,IQNIINDX)
        ENDIF
C
          NBAS = NINT(TIMAX*60./DT)+1
          DTIN=DT
          MAXSTP=NBAS-1
          print*,'DT,TIMAX,MAXSTP=',DT,TIMAX,MAXSTP
C
          CPUTIN=SECOND()
          CALL NLMOD
          CPUTOUT=SECOND()
          print*,'NLMOD CPU TIME=', CPUTOUT-CPUTIN
C
 88888 STOP 88888
       END
```

SENSITIVITY

4

4 Sensitivity Experiment

4.1 Purpose

This chapter describes how to run an adjoint sensitivity experiment. The example we provide here calculates the sensitivity of the 12-h forecast of the temperature in the black box in figure 4.1. The response function can thus be expressed as

$$R(\boldsymbol{x_0}) = T(\boldsymbol{i_1}, \boldsymbol{j_1}, \boldsymbol{k_1}) \qquad k = 10, t = 12h \tag{4.1}$$

where $i_1 = \mathbf{11}$ and $j_1 = \mathbf{19}$.

Given $\boldsymbol{x_0}$, the sensitivity of $\boldsymbol{R}$ with respect to $\boldsymbol{x_0}$ can be obtained by running the forward MM5 for 12 h, which provides the basic state trajectory for the following-up adjoint model integration, i.e., running the adjoint model backward with the "initial" condition at the time t=12 h as

$$x^{adj}(t = 12h) = \frac{\partial R}{\partial x_{\boldsymbol{t = 12h}}} \tag{4.2}$$

where $\boldsymbol{x}^{\boldsymbol{adj}}$ represents all the adjoint model variables at the time t=12 h. From (4.1) we obtain that

$$T^{adj}(t = 12h) = \begin{cases} 1 & k = 10, i = 11, j = 19 \\ 0 & otherwise \end{cases} \tag{4.3}$$

The value of $\boldsymbol{x}^{\boldsymbol{adj}}$(t=0) obtained by such an adjoint model backward integration at time t=0 is the sensitivity field. It was saved as

```
      OPEN(UNIT=91,FILE='GRAD.D',FORM='UNFORMATTED',
     -  STATUS='UNKNOWN')
        WRITE(91) UA,VA,WA,TA,QVA,PPA
        CLOSE (91)
```

in the main routine.

Figure 4.1 (see page 4-15) shows the distribution of the sensitivity fields of R defined in (4.1) with respect to the initial temperature field at $\sigma = 0.85$ (Fig. 4.1a) and a cross-section of the sensitivity with respect to the temperature (Fig. 4.1b), the zonal wind (Fig. 4.1c) and the meridional wind (Fig. 4.1d) along a line AB indicated in Fig. 4.1a.

The job deck, main sensitivity routine, and the forward MM5 model and backward MM5 adjoint model are placed in the subdirectory *main_sensitivity*:

- **main_sensitivity**

```
~fdvar/Tutorial_97/main_sensitivity/sens.deck
~fdvar/Tutorial_97/main_sensitivity/sens.F
~fdvar/Tutorial_97/main_sensitivity/main/nlmod_bas.F
~fdvar/Tutorial_97/main_sensitivity/main/adjmod.F
```

Let's copy the main sensitivity routine here before we display some of the sensitivity fields (see Figure 4.1) and address a few additional issues.

4.2 The main routine of the sensitivity calculation: *sens.F*

```
      PROGRAM MSS_SENS
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C        SENSITIVITY CALCULATION OF A RESPONSE                         C
C        (with large-requirement in disc space)                        C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
#    include <parame.incl>
#    include <parmin.incl>
#    include <param2.incl>
#    include <param3.incl>
#    include <addr0.incl>
#    include <various.incl>
#    include <point2d.incl>
#    include <dusolve1.incl>
#    include <variousn.incl>
#    include <point3d.incl>
#    include <pointbc.incl>
#    include <nonhyd.incl>
#    include <nonhydb.incl>
#    include <bcontrol.incl>
#    include <minim.incl>
#    include <obsdata.incl>
C
```

```
      DIMENSION AUA(MIX, MJX, MKX),AVA(MIX, MJX, MKX),
     -     ATA(MIX, MJX, MKX),AQVA(MIX, MJX, MKX),
     -     AWA(MIXNH,MJXNH,KXP1NH),APPA(MIXNH,MJXNH,MKXNH),
     -    ATGA(MIX,MJX), ARAINC(MIX,MJX), ARAINNC(MIX,MJX)
      DIMENSION AQCA(MIXM, MJXM ,MKXM ), AQRA(MIXM ,MJXM ,MKXM ),
     -     AQIA(MIXIC,MJXIC,MKXIC),AQNIA(MIXIC,MJXIC,MKXIC)
C
C*******************************************************************
C
      COMMON /IEXCMN/ IEXEC(MAXNES)
      COMMON /LRG_PRECIPITATION/LRG_PRE
      DIMENSION IERR(2)
      INTEGER TRIMLEN, MSSPATH_LEN
C
C^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
C
      ICOEFFNT=1
      NCEP_WEIGHT=0
      LRG_PRE=1
C
      MSSPATH = 'TMP_DIR005FEB_TO_SAVE_YOUR_RUN_TIME_FILES/'
      MSSPATH_LEN = TRIMLEN(MSSPATH)
C
      IF(MSSPATH(LEN_MSSPATH:LEN_MSSPATH) .NE. '/') THEN
         LEN_MSSPATH = LEN_MSSPATH + 1
         MSSPATH(LEN_MSSPATH:LEN_MSSPATH)  =  '/'
      ENDIF
C
C^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
C
      READ (7,FDPARAM)
      PRINT FDPARAM
C
C--- INITIALIZE TO ZERO
C
      DO 1 N=1,MAXNES
      DO 1 I=1,IHUGE
         ALLARR(I,N)=0.
    1 CONTINUE
      DO 2 N=1,MAXNES
      DO 2 I=1,MIX
      DO 2 J=1,MJX
         XNUU(I,J,N)=0.
         XMUU(I,J,N)=0.
         XNUT(I,J,N)=0.
         XMUT(I,J,N)=0.
    2 CONTINUE
C
C--- FIRST THINGS FIRST: PRESET ADDRESS OF ALL VARIABLES
C--- FOR ALL NESTS.
C
      NSTTOT=1
      CALL ADDALL
C
```

```
C--- INITIALIZE NUMBER OF ACTIVE NESTS AND TOTAL POSSIBLE
C--- NESTS TO ZERO
C
      DO 4 N=1,NLNES
      DO 4 NN=1,MAXNES
         NUMLV(N,NN)=0
    4 CONTINUE
C
C--- FILL COARSE DOMAIN VARIABLES WITH DOMAIN (1)
C
      CALL ADDRX1C(IAXALL(1,1))
C
C-----SET UP PARAMETERS:
C
      DO 5 NN=1,MAXNES
         IEXEC(NN)=1
         LEVIDN(NN)=0
         NUMNC(NN)=0
    5 CONTINUE
C
      IF(MAXNES.EQ.1)THEN
         PRINT *,
     1           '*******************    ONE DOMAIN ONLY!!! ***********'
      ELSE
        PRINT *,'****************** MULTI LEVEL RUN!!! *******'
        PRINT *,'****************** ',MAXNES,' DOMAINS TOTAL*******'
      ENDIF
C
C--- READ IN PARAMETERS AND NAMELISTS
C
      CALL PARAM(IEXEC)
C
C  READ IN SUBSTRATE TEMPERATURE:
C
      READ(9) TMN
      CALL SKIPF (9,1,IERR)
C
C  READ IN THE BOUNDARY CONDITIONS:
C
      CALL  BDYIN( 9,TBDYBE ,BDYTIM ,IL,JL,IBMOIST)
C
C                obtained values of  UEBC,UEBCT,......
C
C---- READ IN THE COARSE GRID INITIAL CONDITIONS
C
      CALL MRDINIT(11, IL, JL, 0)
C                obtained values of  UA,VA,.....WA
C
      CALL MINIT0(IEXEC,IL,JL)
C
C                obtained values of  2-D fields (not variab
C
      NBAS = NINT(TIMAX*60./DT)+1
      DTIN=DT
      MAXSTP=NBAS-1
      print*,'DT,TIMAX,MAXSTP=',DT,TIMAX,MAXSTP
C
C --- Setup save Basic state Frequency.
C
      DO 2222 IT=1,NBAS
         IBSTEP(IT)=IT-1
 2222 CONTINUE
C
      IBASA=29
      IBASB=30
      CALL ASTART(IBASA, NDIMS)
      CALL ASTART(IBASB, NDIMS)
      IF (IBLTYP(1).EQ.2) THEN
         IBAShir=32
         CALL ASTART(IBAShir,NDIMhir)
      ENDIF
C
#ifdef MSS_OPTION
C
C.....In case there isn't big enough disk space for storing all the BASIC STATE,
C   user can use Mass Storage System (MSS) to save the BASIC STATE
C
C.....Suppose your disk quota limit is 6 Gigbytes,
C   and you can use up to 4 Gigbytes to store the BASIC STATE.
C   then the maximum file size for BASIC STATE A or B is 2 Gigbytes.
C Set MAXIMUM_FILE_SIZE =2000000000
C
C      MAXIMUM_FILE_SIZE = 2500000000
C
#ifdef TEST_MSS
      MAXIMUM_FILE_SIZE = 5000000
#else
      MAXIMUM_FILE_SIZE = 2000000000
#endif
C
C   The BASIC STATE file at one-time-level is NDIMS words which can be calculated as
C
      NDIMS_IN_BYTES = 8 * NDIMS
C
C   The maximum total time-steps for the given disk space is:
C
      MAX_TSTEP = MAXIMUM_FILE_SIZE / NDIMS_IN_BYTES
      PRINT *,'BASIC MAX_TSTEP =',MAX_TSTEP
#endif
C
C call nonlinear model to create the basic state for tangent&adjoint mod
C
      print*,'MIX,MJX,MKX,MIXNH,MJXNH,MKXNH,KXP1NH=',
     -     MIX,MJX,MKX,MIXNH,MJXNH,MKXNH,KXP1NH
      print*,'MIXM,MJXM,MKXM,MIXIC,MJXIC,MKXIC=',
     -     MIXM,MJXM,MKXM,MIXIC,MJXIC,MKXIC
      print*,'NSPGD=',NSPGD
      print*,'call NLMOD'
```

```
c
c sensitivity at optimal IC
c
C
C----If the user wants to calculate the sensitivity with respect to the "Optimal" IC,
C----read in the "Optimal" IC here.
C
c       OPEN (UNIT=91,FILE='OPTIMAL.IC',
c     1     FORM='UNFORMATTED',STATUS='UNKNOWN')
c           READ (91) UA,VA,TA,QVA,PPA,WA,QCA,QRA
c       CLOSE(91)
c
c       CALL REMOVE_NEGATIVE(QVA,QCA,QRA,QIA,QNIA,IEXMS,IICE,
c     -             MIX,MJX,MKX,MIXM,MJXM,MKXM,MIXIC,MJXIC,MKXIC,
c     -             IQINDX,IQCINDX,IQRINDX,IQIINDX,IQNIINDX)
c
C
        CPUTIN=SECOND()
        CALL MSS_NLMOD(MAX_TSTEP,MSS_SAVED_NUMBER,MSSPATH)
        CPUTOUT=SECOND()
        print*,'NLMOD CPU TIME=', CPUTOUT-CPUTIN
c user modification
c
c ----Forcing for starting the adjoint model integration
c
        CALL ZERO_VARIABLE(UA, VA, TA, QVA,PPA, WA,
     -             MIX,MJX,MKX,
     -             QCA, QRA,MIXM, MJXM, MKXM, IEXMS,
     -             QIA,QNIA,MIXIC,MJXIC,MKXIC,IICE,
     -             ITGFLG(1),TGA)
      TA(11,19,10)=1.0
e
c user modification ends
        print*,'call ADJMOD'
        CPUTIN=SECOND()
        CALL MSS_ADJMOD(MAX_TSTEP,MSS_SAVED_NUMBER,MSSPATH)
        CPUTOUT=SECOND()
        print*,'ADJMOD CPU TIME=', CPUTOUT-CPUTIN
C
        OPEN(UNIT=91,FILE='GRAD.D',FORM='UNFORMATTED',
     -      STATUS='UNKNOWN')
        WRITE(91) UA,VA,WA,TA,QVA,PPA,QCA,QRA
        CLOSE (91)
C
 88888 STOP 88888
        END
```

4.3 A more general adjoint sensitivity study

Adjoint sensitivity studies differ in the definition of the response function *R*. A more general definition of *R* can be written as

$$R(x_0) = G(H(x(t_r))) \tag{4.4}$$

where $H(x(t_r))$ is a nonlinear function of the model forecast at time t_r, which is usually called as a forward model operator, and G is a scalar function of $H(x(t_r))$.

In order to calculate the sensitivity of the response function defined in (4.4), users are required to first linearize the subroutine representing the operator H with respect to the model forecast $x(t_r)$:

$$\frac{\partial}{\partial x(t_r)} H(x(t_r)) \tag{4.5}$$

Then, users need to write an adjoint subroutine realizing the transpose of the above-linearized-subroutine:

$$\left(\frac{\partial}{\partial x(t_r)} H(x(t_r))\right)^T \tag{4.6}$$

The value of the gradient of R defined in (4.4) can then be calculated as the result of a series of operators:

$$\nabla_{x_0} R = P_r^T \underbrace{\underbrace{\left(\frac{\partial}{\partial x(t_r)} H(x(t_r))\right)^T}_{\text{(adjoint of } H\text{)}} \frac{\partial}{\partial H(x(t_r))}(G(H(x(t_r))))}_{\text{(forcing term)}} \tag{4.7}$$

where P_r^T is the MM5 adjoint model integrated from time $t = t_r$ to t=0.

To run a sensitivity experiment different from the standard example set in the code, users are required to modify the part which is bold faced in Section 4.2, i. e., substituting

$$x^{adj}(t = t_r) = \left(\frac{\partial}{\partial x(t_r)} H(x(t_r))\right)^T \frac{\partial}{\partial H(x(t_r))} G(H(x(t_r))) \tag{4.8}$$

with (4.2).

If the response function contains more than one-time-level forecast aspects, i. e.

$$R(x_0) = G_1(H_1(x(t_1))) + G_2(H_2(x(t_2))) \tag{4.9}$$

where $t_1 < t_2$. The gradient of the response function becomes the summation of two terms:

$$\nabla_{x_0} R = \nabla_{x_0} R_1 + \nabla_{x_0} R_2 \tag{4.10}$$

where

$$\nabla_{x_0} R_1 = P_{t_1}^T \left(\frac{\partial}{\partial x(t_1)} H_1(x(t_1))\right)^T \frac{\partial}{\partial H_1(x(t_1))} G_1(H_1(x(t_1))) \tag{4.11}$$

$$\nabla_{x_0} R_2 = P_{t_2}^T \left(\frac{\partial}{\partial x(t_2)} H_2(x(t_2))\right)^T \frac{\partial}{\partial H_2(x(t_2))} G_2(H_2(x(t_2))) \tag{4.12}$$

Users can either run two sensitivity jobs and add the two gradient afterward, or run one sensitivity job and obtain the total gradient. For the former, one job runs between the time t_0 and t_1 and the other between the time t_0 and t_2, with the forcing terms in (4.11) and (4.12) being implemented in each of the main programs *sens.F*. For the latter, due to the linearity of the adjoint model, users can obtain the total gradient by integration of the forward MM5 and backward adjoint model only once in the time window t_0 and t_2 with the forcing at time t_2 in (4.12) being added in the main program *sens.F* and the forcing at time t_1 in (4.11) being added to the adjoint variables at time t_1 in the adjoint model; i. e., users now have to edit the file:

~fdvar/Tutorial_97/main_sensitivity/main/adjmod.F.

Therefore, for an adjoint sensitivity calculation, users are required to integrate both the MM5 forward and the MM5 adjoint model in time only once. The computational cost in the adjoint sensitivity calculation is thus much less than a 4D-VAR experiment in which both the MM5 and its adjoint model need to be integrated many times (equal or greater than the number of the iterations). One may run a much larger job (bigger domain or higher resolution) than that for a 4D-VAR experiment. When this is the case, however, the disk space may become a restriction due to the need to save the basic state at every time-step. To solve this problem, an in-job use of mass storage capability is implemented in the standard MM5 adjoint modeling system.

Suppose that the disk quota in a user's working directory is 6 Gigbytes, and out of it he can use up to 4 Gigbytes for storing the basic state. The user should set the parameter

$$MAXIMUNM-FILE-SIZE = 2 \times 10^9$$

since we have to save the basic state at both time levels: t and t-1, MAXIMUM_FILE_SIZE was not set to 4 x 10^9. The program will calculate the maximum total time steps (MAX_TSTEP) to save the basic state simultaneously for the given disk space. In the forward model run, when these maximum total time steps are reached, we write all the basic state to the mass storage, clean the directory, and accumulate the following basic state until reaching the time steps 2xMAX_TSTEP, and so on. When the adjoint model is integrated backward in time, the basic state saved in the mass storage will be read block by block, the newest will be read first and the oldest will be read last. This procedure can be implemented to another facility instead of mass storage.

The job deck *sens.deck* to run sensitivity experiment is listed below:

4.4 The job deck of the sensitivity calculation: sens.deck

```
# QSUB -r SENS              # request name
# QSUB -q reg              # job queue class prem  econ
# QSUB -eo                #
# QSUB -o  sens.out              # stdout and stderr together
# QSUB -lM 8Mw                    # maximum memory
# QSUB -lT 5000                  # time limit
# QSUB                          # no more qsub commands
#
 ja
#
 set echo
#          *******************************************
#          *******    4dvar batch C shell     *******
#          *******************************************
set Data_Dir = ~TUTORIAL/data
 set Work_Dir = ~tmp-s
 set Curt_Dir = `pwd`

# set Recompiled = Yes      # recompile the code
  set Recompiled = No
#
#     type of 4dvar job
#
 set ReStart  = False     # True -- restart minimization run
#
#
#     local namelist values
#
cat >! $Work_Dir/mmlif << EOF
 &FDPARAM
IFVERIGR = 0
 LRG_PRE  = 1,     ; 0,1 large-scale precipitation
 & ;-----------------------------------------------------------------
 &OPARAM                                 ;         <-- MOD2
 ;-----------------------------------------------------------------
 ;    YOU CAN REMOVE THE UNWANTED DATA FROM THE FOLLOWING LISTING
 ;    AND USE THE DEFAULT VALUES DEFINED IN SUBROUTINE 'PARAM'.
 ;-----------------------------------------------------------------
 IFREST = F,     ;RESTART
   IXTIMR  =720,
 LEVIDN  = 0, ; level of nest for each domain
 NUMNC   = 1, ; ID of mother domain for each nest
 IFSAVE = F,      ; SAVE DATA FOR RESTART
   SAVFRQ =180., ;  ... in minutes.
 IFTAPE = 1,      ; OUTPUT FOR GRIN backend
   TAPFRQ =60., ;  ... in minutes.
 IFPRT = 1,       ; do not change
PRTFRQ = 180.,   ; Print output frequency in minutes
 MASCHK = 30,     ; MASS CONSERVATION CHECK (minutes)
 & ;-----------------------------------------------------------------
 &LPARAM
  iactiv=1,0,0,0,0,0,0,0,0,0, ; in case of restart: was this domain active?
```

```
;
;************* start physics options ****************
;
IFRAD    = 0,   ;RADIATION COOLING OF ATMOSPHERE - 0, 1, 2
RADFRQ   = 30., :RADIATION FREQUENCY IN MINUTES
ICUSTB   = 1,   ;STABILITY CHECK FOR CUMULUS PARAM. - 0(no stab. check), 1
IEXICE   = 0,   ;ICE-PHYSICS IN EXPLICIT SCHEME - 0, 1
IFDRY    = 0,   ;FAKE-DRY RUN - 0, 1
IMVDIF   = 0,   ;MOIST VERTICAL DIFFUSION IN CLOUDS - 0, 1
IBMOIST  = 0,   ;BOUNDARY AND INITIAL WATER/ICE SPECIFIED - 0, 1
ICOR3D   = 1,   ;3D CORIOLIS FORCE - 0, 1
IFUPR    = 0,   ;UPPER RADIATIVE BOUNDARY CONDITION - 0, 1
;
IBOUDY = 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, ;BOUNDARY CONDITIONS - 0, 1, 2, 3, 4
IBLTYP = 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, ;PBL TYPE - 0, 1, 2
IDRY   = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;MOIST OR DRY CASE - 0, 1
IMOIST = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;NON-EXPLICIT,  EXPLICIT, -  1, 2
ICUPA  = 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;NONE,KUO,GRELL,AS - 1,2,3,4
ISFFLX = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;SURFACE FLUXES - 0, 1
ITGFLG = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;SURFACE TEMPERATURE - 1, 3
ISFPAR = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;SURFACE CHARACTERISTICS - 0, 1
ICLOUD = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;CLOUD EFFECTS ON RADIATION - 0, 1
ICDCON = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;CONSTANT DRAG COEFFICIENTS - 0, 1
IFSNOW = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;SNOW COVER EFFECTS - 0, 1
IMOIAV = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;VARIABLE MOISTURE AVAILABILITY - 0, 1
IVMIXM = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;VERTICAL MIXING OF MOMENTUM - 0, 1
HYDPRE = 1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,;HYDRO EFFECTS OF LIQ WATER - 0., 1.
IEVAP  = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;EVAP OF CLOUD/RAINWATER - <0, 0, >0
ISHALLO= 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;SHALLOW CONVECTION - 0, 1
;
;************* end physics options ******************
;
;
;
;************* start nesting options ****************
;
nestix = 26,  46,  46,  46,  46,  46,  46,  46,  46,  46, ; domain size I
nestjx = 31,  61,  61,  61,  61,  61,  61,  61,  61,  61, ; domain size J
nesti  =  1,   1,   1,   1,   1,   1,   1,   1,   1,   1, ; start location I
nestj  =  1,   1,   1,   1,   1,   1,   1,   1,   1,   1, ; start location J
xstnes =  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,; domain initiation
xennes =720.,720.,720.,720.,720.,720.,720.,720.,720.,720.,; domain completion
ioverw =  0,   0,   0,   0,   0,   0,   0,   0,   0,   0, ; overwrite domain
;
;************* end nesting options ******************
& ;-------------------------------------------------------------
&PPARAM
TIMAX  = 720.,  ; IN MINUTES. 720=12h, 1440=24h, 2160=36h, 2880=48h
ZZLND  = 0.1,   ; ROUGHNESS LENGTH OVER LAND IN METERS
```

```
ZZWTR  = 0.0001, ; ROUGHNESS LENGTH OVER WATER IN METERS
ALBLND = 0.15,   ; ALBEDO
THINLD = 0.04,   ; SURFACE THERMAL INERTIAL
XMAVA  = 0.3,    ; MOISTURE AVAILABILITY OVER LAND AS A DECIMAL
;                  FRACTION OF ONE
CONF   = 1.0,    ; NON-CONVECTIVE PRECIPITATION SATURATION
TISTEP = 360.,   ; COARSE DOMAIN DT IN MODEL, USE 3*DX
ifeed  = 3,      ; OLD FEEDBACK, NO/LIGHT SMOOTHING IN FEEDBK - 1,2,3
iabsor = 0,      ; SPONGE ON UPPER BOUNDARY - 0,1
& ;--------------------------------------------------------------
 &FPARAM
 & ;--------------------------------------------------------------
EOF
#
###########################################################################
#####################                      ########################
#####################  END USER MODIFICATION  ########################
#####################                      ########################
###########################################################################
#
#     initializations, no user modification required
#
set LETTERS = (A B C D E F G H I J K L M N O P Q R S T U V W X Y Z)
set FDvarUser = ~mesouser/MM5V1/MM5
#
 if ( $Recompiled == Yes ) then
#
  make
ls -l
endif
#
#       Get the input files from MSS
#
rm fort.*
ln -s mmlif            fort.7
 ln -s ~TUTORIAL/FDVAR/data/ehtran.ascii         fort.8
 ln -s $Data_Dir/bdyout                    fort.9
 ln -s $Data_Dir/guess_input_kl4dvar.o       fort.11
# ln -s $Data_Dir/guess_input_kl4dvar        fort.11
 ln -s $Data_Dir/DirObs_kl4dvar            fort.20
 ln -s $Curt_Dir/sens.exe sens.exe

 execute:
#
#     run FDVAR
#
cd $Work_Dir
 timex sens.exe >&! $Curt_Dir/sens.print.out
#
```

The places where users may want to modify are indicated by bold face lines. They are related to where to read in the input files, the choice of physics options, and the setting of the dimension and time step, as well as the place to save the output files.

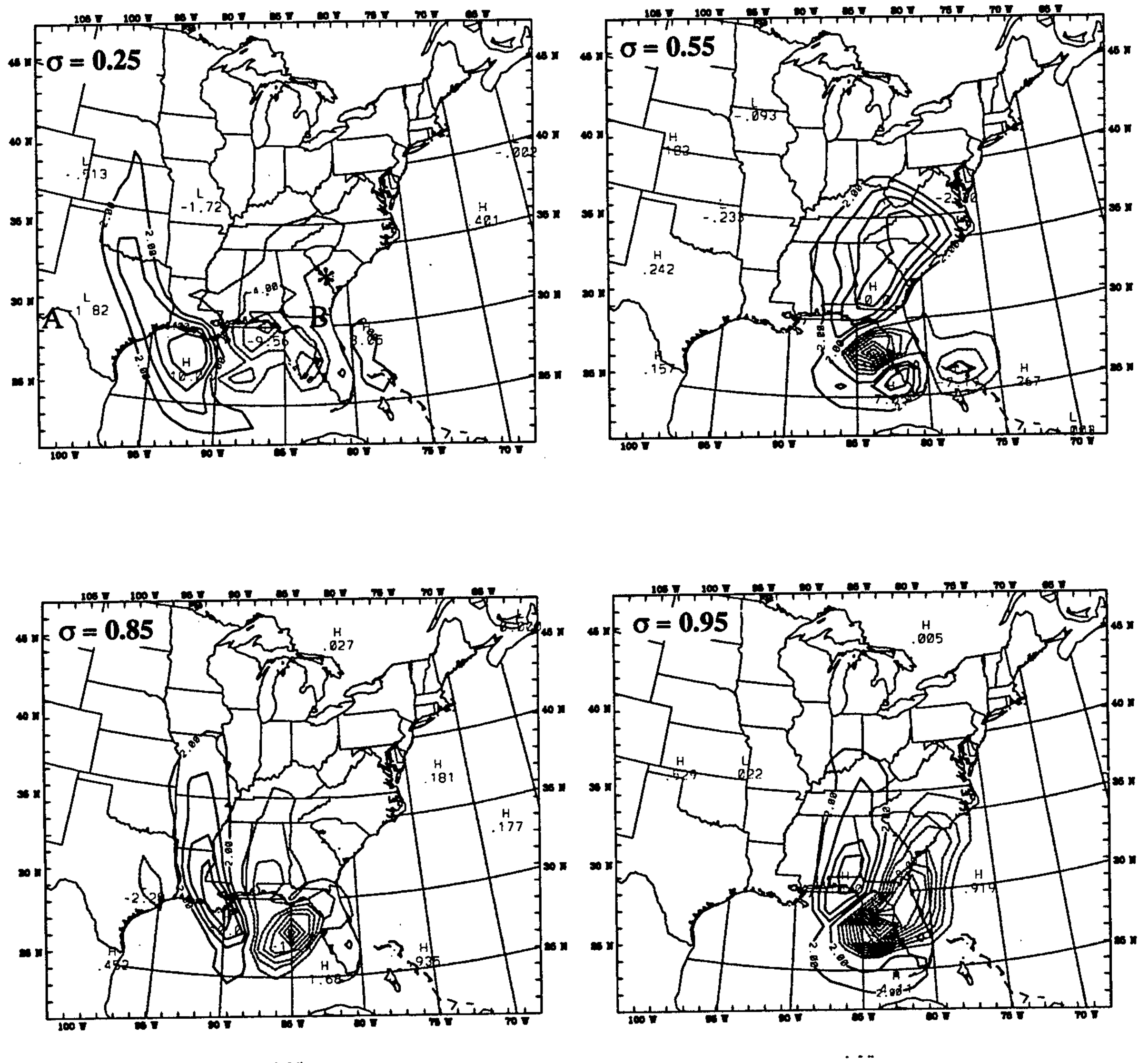


Figure 4.1: (a) The sensitivity of R defined in (4.1) with respect to the initial temperature field field at σ = 0.25, 0.55, 0.85 and 0.95.

4D-VAR

5

5 4D-VAR

5.1 Purpose

The 4D-VAR programs seek the "optimal" IC for a user-pre-defined cost function, which measures the misfit of the model prediction to the given observations in the least-square sense. For different data-assimilation objectives, the cost function to be minimized will be different. Therefore, unlike using MM5 modeling system where users can run the job without knowing many of the details of the model code, the user needs to modify some of the codes in the MM5 4D-VAR system to find the solution of their own data assimilation problem, due to the difference in observations to be assimilated, the definition and use of weighting and scaling, and possible the penalty constraints users may wish to add. In this section, we will describe the 4D-VAR system in detail with the hope that users can not only run the standard 4DVAR experiment, but also run their own 4DVAR experiment with ease.

The job deck, main program, and a few subroutines which may need user interactions for carrying out 4D-VAR experiment are placed in the subdirectory *main_minimization*:

~fdvar/Tutorial_97/main_minimization/fdvar.deck
~fdvar/Tutorial_97/main_minimization/mindry.F
~fdvar/Tutorial_97/main_minimization/main/mm5dry.F
~fdvar/Tutorial_97/main_minimization/main/adjdry.F
~fdvar/Tutorial_97/main_minimization/main/objfun.F
~fdvar/Tutorial_97/main_minimization/main/gradcl.F

The subroutine *mindry.F* is the main driver to carry out the minimization procedure (see Fig. 1 in Zou et al., 1997), *mm5dry.F* and *adjdry.F* are the MM5 and MM5 adjoint model incorporated with the calculation of the cost function and proper forcing terms for the gradient calculation of the cost function (see Fig. 2 in Zou et al., 1997), the subroutine *objfun.F* calls the forward model *mm5dry.F* from the minimization routines in which the control variables is a one-dimensional array, and *gradcl.F* calls the *adjdry.F* to get the scaled gradient value during the minimization process. Before we move on to explain how to run a 4D-VAR experiment, we first mention the physics options available in the MM5 adjoint model system:

5.2 Physics Options in MM5's 4D-VAR System

5.2.1 Cumulus Parameterizations

- None
  Use no cumulus parameterization.
- Anthes-Kuo scheme
  Based on moisture convergence, specified heating profile, moistening dependent upon relative humidity, and mostly applicable to grid size > 30 km.
- Grell scheme
  Stability closure with an updraft and a downdraft.

5.2.2 PBL Schemes

- None
  No surface layer, unrealistic in real-data simulation.
- Bulk PBL
  Suitable for coarse vertical resolution in boundary layer, e.g. > 250m vertical grid size. Two stability regimes.
- High-resolution PBL
  Suitable for higher vertical resolution in boundary layer. Four stability regimes.

5.2.3 Explicit Moisture Schemes

- Dry
  No moisture prediction. Zero water vapor
- Stable Precipitation
  Non-convective precipitation. Large scale saturation removed and rained out immediately. No rain evaporation or explicit cloud prediction.
- Dudhia's microphysics scheme.
  Explicit treatment of cloudwater, rainwater, snow, and ice.

5.2.4 Radiation Schemes

- None
  No mean tendency applied to atmospheric temperature, unrealistic in long-term simulation.
- Simple cooling
  Atmospheric cooling rate depends just on temperature. No cloud interaction or diurnal cycle.
- Surface radiation
  Provide diurnally varying shortwave and longwave flux at the surface for use in the ground energy budget. These fluxes are calculated based on atmospheric column-integrated water and low/middle/high cloud fraction estimated from relative humidity.

5.3 Procedures to run a 4D-VAR experiment

- Edit the file "*configure.user*" to set up the RUNTIME_SYSTEM;
- Edit "*parame.incl*" in the *include* directory to setup your domain size (MIX, MJX, and MKX) , and "*parmin.incl*" in the include directory to setup observation-related parameters and etc. (NOBS_TYPE, NOBS, NOBS, NOBS2, NOBS3, NOBS_TIME, and NBC, see Table 5.1);
- Modify *mindry.F*, *mm5dry.F* and *adjdry.F* according to observations to be assimilated, the preferred weighting and scaling calculation, and additional constraints you wish to add to the cost function;
- Type "make" in the subdirectory *main_minimization* to produce an executable file;
- Modify the job deck *fdvar.deck*, mainly for the input files, output files, physics options, domain size, and time step, and submit the job deck.

The following section contains more details about a few key parameters, expected input and output files, namelist, and definition of script variables:

5.3.1 A few key parameters to be altered for different assimilation experiments:

The standard code is set to run a simple twin experiment. If users wish to run their own data

Table 5-1:

Parameters	Directory/files	Explanation
MIX, MJX, MKX	**~fdvar/Tutorial_97/include/ parame.incl**	**model grid size**
NOBS_TYPE	**~fdvar/Tutorial_97/include/ parmin.incl**	**Number of different types of obs**
NOBS	**same as above**	**Number of total time levels for direct obs**
NOBS2	**same as above**	**Number of total time levels for indirect obs**
NOBS_TIME	**same as above**	**max(NOBS, NOBS2)**
NBC	**same as above**	**Number of LBCs**
TIMAX	**~fdvar/Tutorial_97/ main_minimization/fdvar.deck**	**Length of assimilation window in minute**
TISTEP	**same as above**	**Time step in second**
MIX, MIX, MKX, nestix, nestjx	**same as above**	**Model grid size which should be made consistent with MIX, MJX, and MKX in parame.incl**

assimilation experiment, a few key parameters, which are listed in Table 5.1, need to be modifed according to the job size, resolution, and available observations to be assimilated.

5.3.2 Input and output files:

Input files:

1) Standard MM5 initial file and boundary file:

- InBdy = MM5.BC (standard MM5 LBC input file)
- InMM = MM5.IC (standard MM5 IC file)

2) Observations

- Direct and/or indirect observations

3) Restart files for a restart minimization

- Restart.file: fort.50 (minimization information), ISOL ("optimal" initial perturbation), ICDIR (search direction), ICGRAD (current gradient), IOGRAD (previous gradient)

Output files:

1) ICs at every iteration: ITER.tar

2) Values of the gradient at every iteration: GRAD.tar

3) A print file print.tar containing fdvar.print.out mmlif fdvar.deck acct

4) Restart file: fort.50 (minimization information), ISOL ("optimal" initial perturbation), ICDIR (search direction), ICGRAD (current gradient), IOGRAD (previous gradient)

In-job files:

Weighting, scaling, initial guess, basic states, forcing terms, current and previous gradients, and search direction.

In 4D-VAR, different input and output data are written to separate files, and most files are accessed by specified Fortran unit numbers, which are assigned as follows:

Table 5.2 Shell names, Fortran unit numbers, MSS names and their usages:

Shell names	**Unit numbers**	Description
mmlif	fort.7	Input, namelist file
ehtran	fort.8	Input, emissivity file
InBdy	fort.9	Input, boundary files created by program INTERP
InMM	fort.11	Input, initial files created by program INTERP
DirObs	fort.20	Input, direct observations

Shell names	Unit numbers	Description
	fort.41	MM5 model output file
	fort.50,23,24,25,26	Output, restart files
	fort.59,23,24,25,26	Iutput, restart files
	fort.71,72,...	Input, indirect observation file(s)
(ITER.####)	fort.91	Output, the updated ICs at every iteration*
(GRAD.####)	fort.91	Output, the gradients at every iteration*
(IINIT)	fort.22	Guess IC
(ISOL)	fort.23	Initial perturbation find by the minimization
(IWGT)	fort.21	Diagonal elements of the weighting matrix
(ISCALE)	fort.28	Scaling factor
(ICGRAD)	fort.26	Current gradient (gradient at iteration k)
(IOGRAD)	fort.27	Previous gradient (gradient at iteration k-1)
IFCST	fort.24,34,35, ...	Forcing terms
(ICDIR)	fort.25	Search direction
(IBASA)	fort.33	Basic state at time t
(IBASB)	fort.30	Basic state at time t-1

*For example, ITER.0001 and ITER.0030 represent the ICs obtained at the 1st and 30th iterations and GRAD.0001 and GRAD.0030 represent the corresponding values of the gradient of the cost function with respect to ICs in ITER.0001 and ITER.0030, respectively.

5.3.3 Namelists for user-specified options

A namelist file, called *mmlif*, is created when *fdvar.deck* is executed. In the following, we skip those variables which are also in the MM5 modeling system and describe only those related variables which are added due to the implementation of the 4D-VAR code.

IFVERIGR	=0, for minimization,
	=1, for gradient check only.
SCLTIM1	>0.0, using the forecast difference between time **SCLTIM2** and **SCLTIM1** (in minutes) to calculate the approximated weighting and scaling.
SCLTIM2	**SCLTIM1** is the first forecast ending time, **SCLTIM2** is the second one.

IRESTART	=TRUE, if it is a restart run,
	=FALSE, if it starts from zeroth iteration.
MSSPATH	=MSS path to save the restart files.
OBSTIME	>0, Direct observation is to be assimilated, put the time in minutes
	<0, No direct observation
OBSTIME1	same as OBSTIME except for the first indirect observation.
OBSTIME2	same as OBSTIME except for the second indirect observation.
ITREND	=Maximum iteration number.
MSS_INTV	=Iteration interval to save the minimization restart files.
NOT_UU	=TRUE, Zero out u weighting, i.e. not assimilate direct observation u,
	=FALSE, assimilate u.
NOT_VV	=TRUE, Zero out v weighting, i.e. do not assimilate direct observation v,
	=FALSE, assimilate v.
NOT_TT	=TRUE, Zero out T weighting, i.e. do not assimilate direct observation T,
	=FALSE, assimilate T.
NOT_QQ	=TRUE, Zero out qv weighting, i.e. do not assimilate direct observation qv,
	=FALSE, assimilate qv.
NOT_PP	=TRUE, Zero out p' weighting, i.e. do not assimilate direct observation p',
	=FALSE, assimilate p'.
NOT_WW	=TRUE, Zero out w weighting, i.e., do not assimilate direct observation w,
	=FALSE, assimilate w.
NCEP_WEIGHT	=0, using the difference between two direct observations to calculate approximated weighting and scaling,
	=1, using NCEP's statistic error to calculate weighting and scaling.
ICOEFFNT	=1, user specifies values of the weighting for indirect observations,
	=0, model calculates the weighting factors for indirect observations which balance each term in the cost function
LRG_PRE	=1, include the large-scale precipitation,
	=0, exclude the large-scale precipitation

5.3.4 Script Variables

In this subsection, we briefly describe the script variables used in the *fdvar.deck.*

NCPUS	number of processors (set NCPUS=1, 4D-VAR codes are not multitasked).
ExpName	experiment name used in setting MSS pathname.
InName	input MSS pathname.
InRst	MSS name(s) of model restart files.
RetPd	mass store retention period (days).
recompiled	=yes, recompile the FDVAR code;
	=no, expect an existing executable.
CaseName	MSS pathname for this run.
OutMM	MSS name for output
RESTART	=FALSE: start model run at zero iteration.
	=TRUE: restart model run.
InBdy	MSS name of boundary file.
InMM	MSS name(s) of model input files.
DirObs	MSS name(s) of direct observation file.

5.4 Some Common Errors Associated with 4D-VAR Failure

When a 4D-VAR job is completed, always check for at least the following:

The "**STOP 99999**" print statement indicates that **4D-VAR WAS** completed successfully.

When running a Cray job, check to make sure that the **mswrite** commands were all completed by the shell, and do a **msls** to check that all the files were written to the pathnames you expected.

Check the top of "fdvar.print.out" file to see if the physics options are correctly specified.
If a 4D-VAR job has failed, check these too.

"Read past end-of-file": This is usually followed by a fortran unit number. Check this unit number with Table 5.2 to find out which file the read error is related in the 4D-VAR problem. Check all the **msread** statements in the print out to make sure that the file was read properly from the mass storage. Also check to make sure that the file size is non-zero. Double-check experiment names and MSS pathnames.

"CPU limit exceeded": Increase the amount of time requested in the **QSUB** statements when running the job on a Cray.

"Not enough space": Increase the amount of memory requested in the **QSUB** statements when running on a Cray.

"Unrecognized namelist variable": This usually means there are typoes in the namelist.

Unmatched physics option: for instance, the following should appear in the output:
  STOP SEE ERRORS IN PRINT-OUT

Uncompiled options:
STOP SEE ERRORS IN PRINT-OUT
  If one browses through one's output, one may find things like:
  ERROR: IFRAD=2, OPTION NOT COMPILED
which tells the user the option you choose has not been compiled.

When restarting a job, do not re-compile the code. If you do, do not change anything in the *configure.user* file.

If the job stopped and there was a long list of "CFL>1...", it can mean that the time step (TISTEP in namelist) is too big or there are other hidden bugs.

5.5 A Twin 4D-VAR Experiment:

In order to make a "cheap" (computationally) and "clean" (knowing the answer) test of the MM5 4D-VAR system, we designed a standard twin 4D-VAR experiment in which "observations" are generated by the assimilation model itself. Most of the components in the 4D-VAR experiment could be well tested in such an identical-twin-experiments 4D-VAR framework. The advantage of conducting the twin experiments is that the true solution is known exactly, and the minimum value of J is zero. It thus allows users to test the accuracy of the code, to examine the performance and capability of the minimization procedure, to assess the speed of convergence and to estimate the computational expenses of such an assimilation.

The 4D-VAR identical twin experiment is initialized at 0000 UTC 13 March 1993. The initial condition was based on the NCEP global analysis, following an objective analyses of rawinsonde and surface observations, which were also carried out for the other time levels between 0000 UTC 13 March to 0000 UTC 14 March 1993 at 12-h interval. The 12-h LBCs were obtained by linear interpolation of the analyses. These IC and LBCs were done at 120-km resolution, with a grid size of 26x31. There are 10 evenly-spaced σ levels in the vertical on both horizontal grids.

The cost function is defined as

$$J = \sum_{r=0}^{3} (x(t_r) - x(t_r)^{obs})^T W(x(t_r) - x(t_r)^{obs}. \tag{5.1}$$

where $x^{obs}(t_r)$, r=0,1,2, and 3 is the MM5 analysis at $t = t_0$ (0000 UTC 13 March 1993, the true IC), and the 1-h, 2-h and 3-h model prediction starting from the "true" IC .

Before a minimization procedure is carried out, a gradient check is suggested to make sure that the values of the cost function and the gradient of the cost function with respect to the IC is correctly calculated. Results of such a check for the above twin experiment are shown in Table 5.3 as follows (see (3.17) in Zou et al. 1997).

Table 5-2:

α	$\psi(\alpha)$
10^{-4}	0.1229677822E+01
10^{-5}	0.1062315239E+01
10^{-6}	0.1000022605E+01
10^{-7}	0.1000003295E+01
10^{-8}	0.1000000662E+01
10^{-9}	0.1000000025E+01
10^{-10}	0.1000013606E+01
10^{-11}	0.1000161181E+01
10^{-12}	0.1001126464E+01
10^{-13}	0.1011717534E+01

in which the first column shows the values of α which controls the magnitude of the initial perturbations, the second and the third columns show the values of $\psi(\alpha)$ with and without including the large-scale precipitation, respectively.

The minimization starts from a guess IC which is the analysis at $t = t_0$. We find that the values of both the cost function and the norm of the gradient with respect to the IC decreased 5 and 3 orders of magnitude respectively (Fig. 5.1) in 20 iterations. In order to examine how the difference fields between the updated IC and the true IC and their subsequent forecasts are modified in the process of minimization, we also plotted the errors in the 300-mb wind fields (Fig. 5.2) and temperature fields at σ=0.95 (Fig. 5.3) at the 0th, 2nd, and 5th iteration, respectively. We found that most errors are corrected during the first few iterations.

5.6 4D-VAR Job Deck

```
# QSUB -r FDVAR          # request name
# QSUB -q reg                 # job queue class prem  reg econ
# QSUB -eo                    #
# QSUB -o fdvar.out           # stdout and stderr together
# QSUB -lM 8Mw                # maximum memory
# QSUB -lT 5000               # time limit
# QSUB                        # no more qsub commands
#
 ja
#
 set echo
#        ********************************************
#        *******    4dvar batch C shell     *******
#        ********************************************
#
 cd $TMPDIR
#
 batchname fdvar_batch.out
#
#   how many CRAY CPUs to use to run the model, set to 1 if not multitasking
#
 setenv NCPUS 1
#
set ExpName = TUTORIAL
set InName = /FDVAR/TUTORIAL
set InRst = $ExpName/RESTART
set RetPd = 999
#
#set Recompiled = Yes        # recompile the code
 set Recompiled = No
#
set CaseName = TEST
#
# MSS directory for output file
#
set OutMM = ${ExpName}/${CaseName}
#
#      type of 4dvar job
#
 set RESTART = False      # True -- restart minimization run
#
#      Give the MSS file names of input data
#
# .. Lateral boundary input file:
 set InBdy = ${InName}/MM5.BC
# .. Standard MM5 input file:
 set InMM = ${InName}/MM5.IC
```

```
# .. Direct observed analysis data file (direct access file):
 set DirObs  = ${InName}/DIRECT_OBS
#
 if ( $RESTART == True ) then
# .. Get the files for restart:
   msread fort.59  ${ExpName}/RESTART/Restart.file
   msread fort.23  ${ExpName}/RESTART/ISOL
   msread fort.25  ${ExpName}/RESTART/ICDIR
   msread fort.26  ${ExpName}/RESTART/ICGRAD
   msread fort.27  ${ExpName}/RESTART/IOGRAD
 endif
#
#      local namelist values
#
cat >! mmlif << EOF
 &FDPARAM
 ;
 IFVERIGR = 0,      ; 1 -> GRADIENT CHECK ONLY, 0 -> MINIMIZATION
 ;
 ; if SCLTIM2 > 0.0, using forecasts at SCLTIM1 and SCLTIM2 to calculate
 ; the weighting and scaling:
 SCLTIM1 =   0.0,   ; in minutes
 SCLTIM2 =   0.0,   ; in minutes
 ;
 RESTART = .$RESTART.,      ; .T. -> restart run
 MSSPATH = '${InRst}',   ;MSS path directory for restart files
 ;
 ; The model time in minute when the observations are available
 ;          1   2   3   4   5   6   7   8   9  10  11  12  13
 OBSTIME  =  0.0, 60.,120.,180.,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,
 OBSTIME2 = -9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,
 OBSTIME3 = -9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,-9.9,
 ;
 ITREND  =  5,      ; ending iterations
 MSS_INTV= 5,        ; interval
 ;
 ; Set .T. to select the variables which are not assimilated:
 NOT_UU = .F.,
 NOT_VV = .F.,
 NOT_TT = .F.,
 NOT_QQ = .F.,
  NOT_PP = .F.,
 NOT_WW = .F.,
 NOT_TG = .T.,
 NCEP_WEIGHT = 0,  ; 0 maximum obs diff, 1 NCEP weight
 ICOEFFNT = 1,     ; 1 user-specified weighting for indirect obs, 0 balance weight
```

```
LRG_PRE  = 1,     ; 0,1 large-scale precipitation
& ;-----------------------------------------------------------------
&OPARAM                              ;        <-- MOD2
;-----------------------------------------------------------------
;    YOU CAN REMOVE THE UNWANTED DATA FROM THE FOLLOWING LISTING
;    AND USE THE DEFAULT VALUES DEFINED IN SUBROUTINE 'PARAM'.
;-----------------------------------------------------------------
IFREST = F,     ;RESTART
  IXTIMR  =720,
LEVIDN  = 0, ; level of nest for each domain
NUMNC   = 1, ; ID of mother domain for each nest
IFSAVE = F,      ; SAVE DATA FOR RESTART
  SAVFRQ =180., ;  ... in minutes.
IFTAPE = 1,      ; OUTPUT FOR GRIN backend
  TAPFRQ =60.,  ;  ... in minutes.
IFPRT = 1,       ; do not change
PRTFRQ = 180.,   ; Print output frequency in minutes
MASCHK = 30,     ; MASS CONSERVATION CHECK (minutes)
& ;-----------------------------------------------------------------
&LPARAM
 iactiv=1,0,0,0,0,0,0,0,0,0, ; in case of restart: was this domain active?
;
;************* start physics options ****************
;
IFRAD    = 0,   ;RADIATION COOLING OF ATMOSPHERE - 0, 1, 2
RADFRQ   = 30., :RADIATION FREQUENCY IN MINUTES
ICUSTB   = 1,   ;STABILITY CHECK FOR CUMULUS PARAM. - 0(no stab. check), 1
IEXICE   = 0,   ;ICE-PHYSICS IN EXPLICIT SCHEME - 0, 1
IFDRY    = 0,   ;FAKE-DRY RUN - 0, 1
IMVDIF   = 0,   ;MOIST VERTICAL DIFFUSION IN CLOUDS - 0, 1
IBMOIST  = 0,   ;BOUNDARY AND INITIAL WATER/ICE SPECIFIED - 0, 1
ICOR3D   = 1,   ;3D CORIOLIS FORCE - 0, 1
IFUPR    = 0,   ;UPPER RADIATIVE BOUNDARY CONDITION - 0, 1
;
IBOUDY = 3, 2, 2, 2, 2, 2, 2, 2, 2, 2, ;BOUNDARY CONDITIONS - 0, 1, 2, 3, 4
IBLTYP = 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, ;PBL TYPE - 0, 1, 2
IDRY   = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;MOIST OR DRY CASE - 0, 1
IMOIST = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;NON-EXPLICIT,  EXPLICIT, -  1, 2
ICUPA  = 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;NONE,KUO,GRELL,AS - 1,2,3,4
ISFFLX = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;SURFACE FLUXES - 0, 1
ITGFLG = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;SURFACE TEMPERATURE - 1, 3
ISFPAR = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;SURFACE CHARACTERISTICS - 0, 1
ICLOUD = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;CLOUD EFFECTS ON RADIATION - 0, 1
ICDCON = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;CONSTANT DRAG COEFFICIENTS - 0, 1
IFSNOW = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;SNOW COVER EFFECTS - 0, 1
IMOIAV = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;VARIABLE MOISTURE AVAILABILITY - 0, 1
```

```
IVMIXM = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;VERTICAL MIXING OF MOMENTUM - 0, 1
HYDPRE = 1.,1.,1.,1.,1.,1.,1.,1.,1.,1.,;HYDRO EFFECTS OF LIQ WATER - 0., 1.
IEVAP  = 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ;EVAP OF CLOUD/RAINWATER - <0, 0, >0
ISHALLO= 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ;SHALLOW CONVECTION - 0, 1
;
;************* end physics options ******************
;
;
;
;************* start nesting options ****************
;
nestix = 26,  46,  46,  46,  46,  46,  46,  46,  46,  46, ; domain size I
nestjx = 31,  61,  61,  61,  61,  61,  61,  61,  61,  61, ; domain size J
nesti  =  1,   1,   1,   1,   1,   1,   1,   1,   1,   1, ; start location I
nestj  =  1,   1,   1,   1,   1,   1,   1,   1,   1,   1, ; start location J
xstnes =  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,; domain initiation
xennes =1440.,720.,720.,720.,720.,720.,720.,720.,720.,720.,; domain completion
ioverw =  0,   0,   0,   0,   0,   0,   0,   0,   0,   0, ; overwrite domain
;
;************* end nesting options ******************
& ;------------------------------------------------------------------
&PPARAM
TIMAX  = 180.,   ; IN MINUTES. 720=12h, 1440=24h, 2160=36h, 2880=48h
ZZLND  = 0.1,    ; ROUGHNESS LENGTH OVER LAND IN METERS
ZZWTR  = 0.0001, ; ROUGHNESS LENGTH OVER WATER IN METERS
ALBLND = 0.15,   ; ALBEDO
THINLD = 0.04,   ; SURFACE THERMAL INERTIAL
XMAVA  = 0.3,    ; MOISTURE AVAILABILITY OVER LAND AS A DECIMAL
;                  FRACTION OF ONE
CONF   = 1.0,    ; NON-CONVECTIVE PRECIPITATION SATURATION
TISTEP = 360,    ; COARSE DOMAIN DT IN MODEL, USE 3*DX
ifeed  = 3,      ; OLD FEEDBACK, NO/LIGHT SMOOTHING IN FEEDBK - 1,2,3
iabsor = 0,      ; SPONGE ON UPPER BOUNDARY - 0,1
& ;------------------------------------------------------------------
&FPARAM
& ;------------------------------------------------------------------
EOF
#
##########################################################################
####################   END USER MODIFICATION   ##################
##########################################################################
#
#     initializations, no user modification required
#
set LETTERS = (A B C D E F G H I J K L M N O P Q R S T U V W X Y Z)
set FDvarUser = ~mesouser/MM5V1/MM5
```

```
#
if ( $Recompiled == Yes ) then
make
 mswrite -t 1000 fdvar.exe $OutMM/fdvar.exe
 ls -l
else
 msread fdvar.exe $OutMM/fdvar.exe
 chmod +x fdvar.exe
endif
#
#       Get the input files from MSS
#
 if( ! -e InBdy ) msread InBdy $InBdy
 if( ! -e InMM ) msread InMM $InMM
 if( ! -e DirObs  ) msread DirObs  $DirObs
#
#     set up fortran input files for 4DVAR
#
 rm fort.*
 ln -s mmlif              fort.7
 ln -s ${FDvarUser}/ehtran   fort.8
 ln -s InBdy              fort.9
 ln -s InMM               fort.11
 ln -s DirObs              fort.20
#ln -s InDirPW              fort.71
#ln -s InDirRN              fort.77
#
 execute:
#
#     run FDVAR
#
 date
 fdvar.exe >&! fdvar.print.out
#
 ja -st >! acct
 cat acct >> fdvar.print.out
 cat fdvar.print.out
#
 tar -cvf GRAD.tar GRAD.*
   mswrite -t 900 GRAD.tar $OutMM/GRAD.tar
#
   tar -cvf ITER.tar ITER.* LAST.*
   mswrite -t 900 ITER.tar $OutMM/ITER.tar
#
   tar -cvf print.tar fdvar.print.out mmlif fdvar.deck acct
   mswrite -t 900 print.tar $OutMM/print.tar
```

```
#
  rm GRAD.tar ITER.tar print.tar
```

5.7 The main 4D-VAR program: *mindry.F*

```
      PROGRAM MINDRY
      SAVE
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C   L-BFGS minimization with a nonhydrostatic version of mm5          C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
#    include <parame.incl>
#    include <parmin.incl>
#    include <param2.incl>
#    include <param3.incl>
#    include <addr0.incl>
#    include <various.incl>
#    include <dusolve1.incl>
#    include <variousn.incl>
#    include <point3d.incl>
#    include <point2d.incl>
#    include <pointbc.incl>
#    include <nonhyd.incl>
#    include <nonhydb.incl>
#    include <bcontrol.incl>
#    include <minim.incl>
C
      DIMENSION AUA (MIX,MJX,MKX),AVA (MIX,MJX,MKX),ATA(MIX,MJX,MKX   ),
     -     AQVA(MIX,MJX,MKX),APPA(MIX,MJX,MKX),AWA(MIX,MJX,KXP1NH),
     -     AQCA(MIXM ,MJXM ,MKXM ),AQRA (MIXM ,MJXM ,MKXM ),
     -     AQIA(MIXIC,MJXIC,MKXIC),AQNIA(MIXIC,MJXIC,MKXIC),
     -     ATGA(MIX,MJX)
      DIMENSION IFCST(NOBS_TYPE)
      COMMON /LRG_PRECIPITATION/LRG_PRE
C
      COMMON /SCRTCH/XSBB(NDIMSBB),XSS(NDIMS),
     -     ZZZ(NSCRTCH-NDIMSBB-NDIMS)
C
C L-BFGS
      LOGICAL FINISH
C MM5
        INTEGER IEXEC(MAXNES)
        INTEGER TRIMLEN, LEN_MSSPATH
C OUTPUT FILENAME
        CHARACTER FILENAME*9
#    include <obsdata.incl>
C
C^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
C
        ICOEFFNT=1
        NCEP_WEIGHT=0
        LRG_PRE=1
        MSSPATH = 'TMP_DIRECTORY_TO_SAVE_YOUR_RUN_TIME_FILES/'
        READ (7,FDPARAM)
        PRINT FDPARAM
C
      LEN_MSSPATH = TRIMLEN(MSSPATH)
      IF(MSSPATH(LEN_MSSPATH:LEN_MSSPATH) .NE. '/') THEN
         LEN_MSSPATH = LEN_MSSPATH + 1
         MSSPATH(LEN_MSSPATH:LEN_MSSPATH)  =  '/'
      ENDIF
C
C--- INITIALIZE TO ZERO
C
      DO 1 N=1,MAXNES
      DO 1 I=1,IHUGE
         ALLARR(I,N)=0.
  1   CONTINUE
      DO 2 N=1,MAXNES
      DO 2 I=1,MIX
      DO 2 J=1,MJX
         XNUU(I,J,N)=0.
         XMUU(I,J,N)=0.
         XNUT(I,J,N)=0.
         XMUT(I,J,N)=0.
  2   CONTINUE
C
C--- FIRST THINGS FIRST: PRESET ADDRESS OF ALL VARIABLES
C--- FOR ALL NESTS.
C
        NSTTOT=1
        CALL ADDALL
C
C--- INITIALIZE NUMBER OF ACTIVE NESTS AND TOTAL POSSIBLE
C--- NESTS TO ZERO
C
      DO 4 N=1,NLNES
      DO 4 NN=1,MAXNES
         NUMLV(N,NN)=0
  4   CONTINUE
C
C--- FILL COARSE DOMAIN VARIABLES WITH DOMAIN (1)
C
        CALL ADDRX1C(IAXALL(1,1))
C
C-----SET UP PARAMETERS:
C
C--- FIRST THINGS FIRST: PRESET ADDRESS OF ALL VARIABLES
C--- FOR ALL NESTS.
C
      NSTTOT=1
      CALL ADDALL
C
C--- INITIALIZE NUMBER OF ACTIVE NESTS AND TOTAL POSSIBLE
C--- NESTS TO ZERO
C
      DO 4 N=1,NLNES
      DO 4 NN=1,MAXNES
         NUMLV(N,NN)=0
    4 CONTINUE
C
C--- FILL COARSE DOMAIN VARIABLES WITH DOMAIN (1)
C
      CALL ADDRX1C(IAXALL(1,1))
C
C-----SET UP PARAMETERS:
C
      DO 5 NN=1,MAXNES
         IEXEC(NN)=1
         LEVIDN(NN)=0
         NUMNC(NN)=0
    5 CONTINUE
C
      IF(MAXNES.EQ.1)THEN
         PRINT *,
     1         '*******************    ONE DOMAIN ONLY!!! ***********'
      ELSE
         PRINT *,'****************** MULTI LEVEL RUN!!! *******'
         PRINT *,'****************** ',MAXNES,' DOMAINS TOTAL*******'
      ENDIF
C
C--- READ IN PARAMETERS AND NAMELISTS
C
      CALL PARAM(IEXEC)
C
C  READ IN SUBSTRATE TEMPERATURE:
C
      READ(9) TMN
      CALL SKIPF (9,1,IERR)
C
C  READ IN THE BOUNDARY CONDITIONS:
C
      CALL  BDYIN( 9,TBDYBE ,BDYTIM ,IL,JL,IBMOIST)
C
C                 obtained values of  UEBC,UEBCT,......
C
C---- READ IN THE COARSE GRID INITIAL CONDITIONS
C
      CALL MRDINIT(11,        IL,  JL,  0)
C                             ^     ^    ^    ^
C                             |     |    | ---- 1=READ HEADER ON DATA SET
C                             |     ------------ I,J SIZE OF THIS DOMAIN
C                             |---------------- UNIT OF INITIAL CONDITION FILE
C
C                 obtained values of  UA,VA,.....WA
C
      CALL MINIT0(IEXEC,IL,JL)
C
C                 obtained values of  2-D fields (not variab
C
      NBAS = NINT(TIMAX*60./DT)+1
      DTIN=DT
      EXTIME = 0.
      DTINC = 0.
c Total time steps in assimilation wind TIMAX (in minute)
      NBAS = NINT(TIMAX*60./DT)+1
      PRINT *,'  == THE NUMBER OF BASIC STATES NBAS =',NBAS,' =='
      MAXSTP=NBAS-1
c sequential data files
C
C----Define observation file(s) unit.
C
C----IOBS(1) for direct observation
C----IOBS(2)......for other indirect observations.
C
         IOBS(1)=20
C
C----Define weighting unit.
C
      IWGT=21
C
C----Define guess field's unit.
C
      IINIT=22
C
C----Define perturbation solution's unit.
C
      ISOL=23
C
```

```
C----Define units for gradients and search Direction.
C
          ICDIR=25
          ICGRAD=26
          IOGRAD=27
C
C----Define scaling unit.
C
          ISCALE=28
c direct access data files
C
C----Define units for basic state file(s).
C
           IBASB=30
           IBASA= 33
C
C----Define units for the forcing terms.
C
   IFCST(1)=24
           NDIM2D=MIX*MJX
           CALL ASTART(IBASB, NDIMS)
           CALL ASTART(IBASA, NDIMS)
           CALL ASTART(IFCST(1), NDIMSS)
           CALL ASTART(IFCST(2),NDIMSr)
           CALL ASTART(IFCST(3),NDIM2D)
          LENGTH_OF_SOUND=MIX*MJX*(7*MKX+1)
           CALL ASTART(91, LENGTH_OF_SOUND)

c Basic state update frequency
          DO IT=1,NBAS
               IBSTEP(IT)=IT-1
          ENDDO
c Time levels of obs.
c direct OBS
          DO 2221 IT=1,NOBS
               ITSTEP(1,IT) = NINT(OBSTIME(IT)*60./DTIN)
  2221    CONTINUE

C
c--- Calculate weighting and scaling from either two given analyses or two forecasts
c
          CALL TRANSF( 1,XSBB,NDIMSBB, UA, VA, TA, QVA,PPA, WA,
     -              MIX,MJX,MKX,
     -              QCA, QRA,MIXM, MJXM, MKXM, IEXMS,
     -              QIA,QNIA,MIXIC,MJXIC,MKXIC,IICE)
c
             CALL APUT(IINIT,1,NDIMSBB,XSBB)
C
             IF (SCLTIM2.LE.0.0) THEN
C .. use 2 time-level all-variable direct observations (IOBS(1)) to
c   calculate the scaling factor:
C
C----Get Direct Obsevation(s).
C
              print*,'NOBS=',NOBS,' NDIMOBSS=',NDIMOBSS
              CALL AGET(IOBS(1),1,NDIMOBSS,XSBB)
              CALL AGET(IOBS(1),(NOBS-1)*NDIMOBSS+1,NDIMOBSS,XSS)
C
             ELSE
C .. use the SCLTIM2 minutes model forecasts to calculate the weighting and scale
```

```
C
            LENGTH1= NINT((SCLTIM1*60.)/DT)
            LENGTH2= NINT((SCLTIM2*60.)/DT)
            IF (LENGTH1.EQ.0) THEN
              CALL AGET(IINIT,1,NDIMSBB,XSBB)
            ELSE
              CALL NLMOD(LENGTH1,DTIN)
              CALL TRANSF( 1,XSBB  ,NDIMSBB, UA, VA, TA, QVA,PPA, WA,
     -           MIX,MJX,MKX,
     -           QCA, QRA,MIXM, MJXM, MKXM, IEXMS,
     -           QIA,QNIA,MIXIC,MJXIC,MKXIC,IICE)
         ENDIF
            LENGTH2= NINT((SCLTIM2*60.)/DT)
            CALL NLMOD(LENGTH2,DTIN)
            CALL TRANSF( 1,XSS  ,NDIMSS, UA, VA, TA, QVA,PPA, WA,
     -           MIX,MJX,MKX,
     -           QCA, QRA,MIXM, MJXM, MKXM, IEXMS,
     -           QIA,QNIA,MIXIC,MJXIC,MKXIC,IICE)
         ENDIF
C
C (1) scaling factor
c Maximum difference (Max(abs(u1-u2)))
           CALL MAXDIFF(XSBB,NDIMSBB,XSS,NDIMS,
     -        UMAX,VMAX,TMAX,QVMAX,PPMAX,WMAX,
     -        MIX,MJX,MKX,
     -        QCMAX, QRMAX,MIXM, MJXM, MKXM, IEXMS,
     -        QIMAX,QNIMAX,MIXIC,MJXIC,MKXIC,IICE)
C
         PRINT 902
  902      FORMAT(/2X,'SCALE OF THE VARIABLES:'/
     -      ' K ',' SIGMA ','    UMAX    ','    VMAX    ',
     -      '     TMAX    ','    QMAX    ','    PPMAX   ',
     -      '     WMAX    ','    QCMAX   ','    QRMAX   ',
     -      '     QIMAX   ','   QNIMAX   ')
           DO K = 1,MKX
           PRINT 903,K,A(K),UMAX(K),VMAX(K),TMAX(K),QVMAX(K),PPMAX(K),
     -      WMAX(K),QCMAX(K),QRMAX(K),QIMAX(K),QNIMAX(K)
  903         FORMAT(I3,F8.3,10E14.5)
           END DO
C .. Square of *MAX and placed in XSBB:
         CALL SCALING(XSBB,NDIMSBB,UMAX,VMAX,TMAX,QVMAX,PPMAX,WMAX,
     -        QCMAX,QRMAX,QIMAX,QNIMAX,MIX,MJX,MKX,IEXMS,IICE)
c
         DO 913 L=1,NDIMSBB
            XSBB(L)=SQRT(XSBB(L))/2.0
  913    CONTINUE
           CALL APUT(ISCALE,1,NDIMSBB,XSBB)

C ---------------------------------------------------------------------
         DO 912 L=1,NDIMSBB
              XSBB(L) = 2.0*XSBB(L)
              IF (L.LE.NDIMSS) XSS(L)= XSBB(L)
  912    CONTINUE
         IF (NCEP_WEIGHT.EQ.1) THEN
C
C  .. (b) Get the OBS at the first time period and transfer
C       to UA, VA, TA, QVA, PPA,...:
         CALL AGET(IINIT,1,NDIMSBB,XSBB)
         CALL TRANSF(-1,XSBB,NDIMSBB, UA, VA, TA, QVA,PPA, WA,
     -            MIX,MJX,MKX,
     -            QCA, QRA,MIXM, MJXM, MKXM, IEXMS,
     -            QIA,QNIA,MIXIC,MJXIC,MKXIC,IICE)
c
         CALL TRANSF(-1,XSS(1),NDIMSS ,AUA, AVA, ATA, AQVA, APPA, AWA,
     -          MIX,MJX,MKX,
     -          AQCA, AQRA,MIXM, MJXM, MKXM, IEXMS,
     -          AQIA,AQNIA,MIXIC,MJXIC,MKXIC,IICE)
C  .. Calculate the weighting for wind, temperature and moisture:
              CALL WEIGHT(MIX,MJX,MKX,PSA,A,PTOP,PPA,TA,QVA,AUA,AVA,ATA,AQVA)
C
C  .. (c) transfer to XSS and calculate the weightings = 1.0/(MAX)^2:
C
         CALL TRANSF( 1,XSS(1),NDIMSS ,AUA, AVA, ATA, AQVA, APPA, AWA,
     -          MIX,MJX,MKX,
     -          AQCA, AQRA,MIXM, MJXM, MKXM, IEXMS,
     -          AQIA,AQNIA,MIXIC,MJXIC,MKXIC,IICE)
C
            ENDIF
c
         DO 922 L=1,NDIMSS
            XSS(L)= 1.0/(XSS(L)*XSS(L))
  922    CONTINUE
C
         CALL TRANSF(-1,XSS(1),NDIMSS ,AUA, AVA, ATA, AQVA, APPA, AWA,
     -         MIX,MJX,MKX,
     -         AQCA, AQRA,MIXM, MJXM, MKXM, IEXMS,
     -         AQIA,AQNIA,MIXIC,MJXIC,MKXIC,IICE)
C
C
C----Take away some of direct observations by zeroing corresponding weighting.
C
c  .. (d) zero-out some weighting coef.
            DO 500 J=1,MJX
            DO 500 I=1,MIX
           DO 500 K=1,MKX
             IF (K.LE.4) THEN
                AQVA(I,J,K)=0.
             ENDIF
C
             IF (NOT_UU) AUA (I,J,K) = 0.
             IF (NOT_VV) AVA (I,J,K) = 0.
             IF (NOT_TT) ATA (I,J,K) = 0.
             IF (NOT_QQ) AQVA(I,J,K) = 0.
             IF (NOT_PP) APPA(I,J,K) = 0.
             IF(K.NE.1 .OR. NOT_WW) AWA(I,J,K) = 0.
  500    CONTINUE
c
            DO K=1,MKX
             I=MIX
             DO J=1,MJX-1
               AWA (I,J,K) = 0.
               ATA (I,J,K) = 0.
               AQVA(I,J,K) = 0.
               APPA(I,J,K) = 0.
             ENDDO
             J=MJX
             DO I=1,MIX
               AWA (I,J,K) = 0.
               ATA (I,J,K) = 0.
               AQVA(I,J,K) = 0.
               APPA(I,J,K) = 0.
             ENDDO
            ENDDO
c
            CALL TRANSF( 1,XSS(1),NDIMSS ,AUA, AVA, ATA, AQVA, APPA, AWA,
     -            MIX,MJX,MKX,
     -            AQCA, AQRA,MIXM, MJXM, MKXM, IEXMS,
     -            AQIA,AQNIA,MIXIC,MJXIC,MKXIC,IICE)
```

```
C .. store in unit IWGT:
              DO 1391 I=1,NOBS
              CALL APUT(IWGT,(I-1)*NDIMSS+1,NDIMSS,XSS)
           1391 CONTINUE
C
c .. (e) PRINT WEIGHTINGS: ............................................
              CALL PRNWGT(MIX,MJX,MKX, AUA, AVA, ATA, AQVA, APPA, AWA, ATGA)
C
C ---------------------------------------------------------------------
          IF (.NOT.RESTART) THEN
C
          DO 11 L=1,NDIMSBB
              XSBB(L)=0.
   11     CONTINUE
              CALL APUT(ISOL, 1,NDIMSBB,XSBB)
C
          IF (IFVERIGR.EQ.1) THEN
C gradient verification
                CALL VERIGR(IINIT,ISOL,IFCST,ISCALE,
        1          ICGRAD,IOGRAD)
                STOP 'Gradient check finished'
          ENDIF
              CALL OBJFUN(VAL,IINIT,ISOL,IFCST)
              CALL GRADCL(IFCST,ISCALE,ICGRAD,IOGRAD,0)
C
              CALL AGET(ICGRAD, 1,NDIMSBB,XSBB)
c
             CALL TRANSF(-1,XSBB,NDIMSBB,AUA,AVA,ATA,AQVA,APPA,AWA,
     -                  MIX,MJX,MKX,
     -                  AQCA, AQRA,MIXM, MJXM, MKXM, IEXMS,
     -                  AQIA,AQNIA,MIXIC,MJXIC,MKXIC,IICE)
c
          OPEN(UNIT=91,FILE='GRAD.0000',FORM='UNFORMATTED',
     -       STATUS='UNKNOWN')
          WRITE(91) AUA,AVA,ATA,AQVA,APPA,AWA,AQCA, AQRA
          CLOSE (91)
C
C----CPP options if your want to save your data to NCAR's MSS.
C
#ifdef MSS_OPTION
              MSSNAME = MSSPATH(1:LEN_MSSPATH)//'GRAD.0000'
              PRINT *,'MSSNAME=',MSSNAME
              CALL WRITE_MSS('GRAD.0000',MSSNAME)
#endif
             ENDIF
C
             IFLAG =0
             NFUN  =1
             IPOINT=0
             FINISH=.FALSE.
C
C .. the number of iterations for minimization:
          ITERMX = ITREND
          ITRBEG = 1
          IF (RESTART) THEN
C
              ICOEFFNT = 1
              CALL RCVR_RST(ITRBEG,ITROUT1,ITROUT2,
        1         STP,VAL,EPS0,IFLAG,NFUN,FINISH,IPOINT, 59)
C
          ENDIF
             CALL LMPRINT(0,NFUN,VAL,ISOL,ICGRAD,1.,FINISH)
C
```

```
C
C----Start iterative procedure of minimization.
C
        DO 888 ITER=ITRBEG,ITERMX
            print*,'iteration ',ITER
            CALL LMDIR(ICDIR,STP,ICGRAD,IOGRAD,ITER,IPOINT,IFLAG)
            IF(IFLAG .LT. 0) GOTO 150
            CALL LMSTEP(ITER,ISOL,VAL,ICDIR,STP,ICGRAD,IOGRAD,IFCST,
     1        ISCALE,IINIT,ITYP,IFLAG,NNFUN,IPOINT)
            IF(IFLAG .LT. 0) GOTO 150
            NFUN=NFUN+NNFUN
            CALL LMCHECK(ICGRAD,ISOL,EPS0,FINISH)
            CALL LMPRINT(ITER,NFUN,VAL,ISOL,ICGRAD,STP,FINISH)
            IF(FINISH) GOTO 150
c
C .. the number of iterations for the first time RESULTS saving:
C
C----CPP options if your want to save your data to NCAR's MSS.
C
#ifdef MSS_OPTION
        IF ((MOD(ITER,MSS_INTV).EQ.0).OR.
     -    (ITER.EQ.1).OR.(ITER.EQ.ITERMX)) THEN
        CALL SAVE_RST(ITER,STP,VAL,EPS0,IFLAG,NFUN,FINISH,IPOINT,
     -          50,ISOL,ICDIR,ICGRAD,IOGRAD,MSSPATH)
            ENDIF
#endif
C
        CALL AGET(ISOL,1,NDIMSBB,XSBB)
        CALL AGET(ISCALE,1,NDIMSBB,XSS)
        DO 11340 L=1,NDIMSBB
             XSBB(L)=XSBB(L)*XSS(L)
 11340   CONTINUE
        CALL AGET(IINIT,1,NDIMSBB,XSS)
        DO 11320 L=1,NDIMSBB
             XSBB(L)=XSBB(L)+XSS(L)
 11320   CONTINUE
C
        CALL TRANSF(-1,XSBB,NDIMSBB,AUA,AVA,ATA,AQVA,APPA,AWA,
     -          MIX,MJX,MKX,
     -          AQCA, AQRA,MIXM, MJXM, MKXM, IEXMS,
     -          AQIA,AQNIA,MIXIC,MJXIC,MKXIC,IICE)
C
C----Output the minimization file.
C
        WRITE(FITNAME(1:9),'(A5,I4.4)') 'ITER.',ITER
        OPEN (UNIT=91,FILE=FITNAME,
     1      FORM='UNFORMATTED',STATUS='UNKNOWN')
        WRITE(91) AUA,AVA,ATA,AQVA,APPA,AWA
        CLOSE(91)
c
c save gradient
C
        CALL AGET(ICGRAD, 1,NDIMSBB,XSBB)
        CALL TRANSF(-1,XSBB,NDIMSBB,AUA,AVA,ATA,AQVA,APPA,AWA,
     -          MIX,MJX,MKX,
     -          AQCA, AQRA,MIXM, MJXM, MKXM, IEXMS,
     -          AQIA,AQNIA,MIXIC,MJXIC,MKXIC,IICE)
C
C----Output the gradient file.
C
        WRITE(FITNAME(1:9),'(A5,I4.4)') 'GRAD.',ITER
```

```
      OPEN (UNIT=91,FILE=FITNAME,
     1     FORM='UNFORMATTED',STATUS='UNKNOWN')
      WRITE(91) AUA,AVA,ATA,AQVA,APPA,AWA
      CLOSE (91)
 888  CONTINUE
 150  CONTINUE
        ILAST=ITER
        IF(ITER.GT.ITERMX) ILAST=ITERMX
        CALL AGET(ISOL,1,NDIMSBB,XSBB)
        CALL AGET(ISCALE,1,NDIMSBB,XSS)
        DO 1344 L=1,NDIMSBB
          XSBB(L)=XSBB(L)*XSS(L)
 1344 CONTINUE
      CALL AGET(IINIT,1,NDIMSBB,XSS)
      DO 1322 L=1,NDIMSBB
          XSBB(L)=XSBB(L)+XSS(L)
 1322 CONTINUE
      CALL TRANSF(-1,XSBB,NDIMSBB,AUA,AVA,ATA,AQVA,APPA,AWA,
     -           MIX,MJX,MKX,
     -           AQCA, AQRA,MIXM, MJXM, MKXM, IEXMS,
     -           AQIA,AQNIA,MIXIC,MJXIC,MKXIC,IICE)
C
C----Output the last minimization file.
C
      WRITE(FITNAME(1:9),'(A5,I4.4)') 'LAST.',ILAST
      OPEN (UNIT=91,FILE=FITNAME,
     1   FORM='UNFORMATTED',STATUS='UNKNOWN')
      WRITE(91) AUA,AVA,ATA,AQVA,APPA,AWA
      CLOSE(91)
c
 99000 CONTINUE
      STOP
      END
```

5.8 The MM5 forward model subroutine incorporated with the calculation of $J(x_0)$ (cost function) and $\frac{\partial J}{\partial x(t)}$ (forcing terms): *mm5dry.F*

```
C
        SUBROUTINE MM5DRY(TVAL,IOUT)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C TVAL --- value of the cost function
C IOUT(i) --- unit to write out the forcing term corresponding IOBS(i)
C
C
C input: initial conditions:  UA,VA,TA,QVA,PPA,WA
C      boundary conditions: FSBC,FNBC,FWBC,FEBC,FSBTC,FNBTC,FWBTC,FEBT
C F represents U,V,T,Q,PP,W
C output: forecasted model state: UA,VA,TA,QVA,PPA,WA
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
#     include <parame.incl>
#     include <parmin.incl>
#     include <param2.incl>
#     include <param3.incl>
#     include <bcontrol.incl>
#     include <dusolve1.incl>
#     include <addr0.incl>
#     include <various.incl>
#    include <minim.incl>
#    include <point3d.incl>
#    include <point2d.incl>
#    include <nonhyd.incl>
#    include <pointbc.incl>
#    include <nonhydb.incl>
C
C*********************************************************************
C
          DIMENSION AUA(MIX,MJX,MKX),AVA(MIX,MJX,MKX),ATA(MIX,MJX,MKX),
     -       AQCA(MIXM , MJXM,MKXM ), AQRA(MIXM , MJXM, MKXM),
     -       AQIA(MIXIC,MJXIC,MKXIC),AQNIA(MIXIC,MJXIC,MKXIC),
     -       ATGA(MIX  ,MJX  ),
     -       AQVA(MIX,MJX,MKX),APPA(MIX,MJX,MKX),AWA(MIX,MJX,KXP1NH)
          COMMON /SCRTCH/XSB(NDIMSB),XS(NDIMS),
     1           ZZZ(NSCRTCH-NDIMSB-NDIMS)
          DIMENSION SATBRT1(MJX)
          DIMENSION VAL(NOBS_TYPE),IOUT(NOBS_TYPE)
C
          INTEGER IEXEC(MAXNES)
          CHARACTER FILENAME*9
C
C initialize
C
          DO I=1,NOBS_TYPE
             VAL(I)=0.
             IDCNT(I)=1
          ENDDO
          DT=DTIN
          IBCNT=1
C
C  Initialize the time of boundary condition:
C
          NB = 1
          BDYTIM = 0.0
C  Initialize the counter KTAU and XTIME:
C
       KTAU  = 0
       XTIME = 0.0
C
C .. RAINC, RAINNC must be initialized (especially for rainfall assimilation
          DO 872 J=1,MJX
          DO 872 I=1,MIX
             RAINC(I,J)=0.
             RAINNC(I,J)=0.
  872     CONTINUE
C
          DO 5 NN=1,MAXNES
             IEXEC(NN)=1
  5       CONTINUE
C
C--- INITIALIZE FROM ZERO
C
          CALL MINIT(1)
          IF (ITGFLG(1) .NE. 1) GO TO 630
          CALL SOLAR1(XTIME,1)
  630     CONTINUE
C
C-----Re-initialize output time:
C
c    SAVTIM=SAVFRQ -0.000001
c    TAPTIM=TAPFRQ -0.000001
c    PRTTIM=PRTFRQ -0.000001
          DECTIM=(1440.-GMT*60.)-0.000001
```

```
C
c                 input: UA,VA
      CALL WBC
C                 output: WA at top and bottom
C
c                 input: UA,...
      CALL UA2UEB
C                 output: UEB,...
      WRITE (FILENAME,'(A6,I3.3)') 'LBCT_.',NB
      CALL BDY_STORE(-1,FILENAME,NB,
     1         UWBT, UEBT, USBT, UNBT,
     2         VWBT, VEBT, VSBT, VNBT,
     3         TWBT, TEBT, TSBT, TNBT,
     4         WWBT, WEBT, WSBT, WNBT,
     5         QWBT, QEBT, QSBT, QNBT,
     6        PPWBT, PPEBT, PPSBT, PPNBT,
     7        QCWBT, QCEBT, QCSBT, QCNBT,
     8        QRWBT, QREBT, QRSBT, QRNBT,
     9        QIWBT, QIEBT, QISBT, QINBT,
     &        QNIWBT,QNIEBT,QNISBT,QNINBT,
     1  MIX, MJX, MKX,MIXNH, MJXNH, MKXNH,KXP1NH, NSPGD,
     2  MIXM, MJXM,MKXM,MIXIC, MJXIC, MKXIC,      NSPGX,
     3  IEXMS,IICE)
c
      WRITE (FILENAME,'(A6,I3.3)') 'LBC__.',NB
      CALL BDY_STORE( 1,FILENAME,NB,
     1         UWB, UEB, USB, UNB,
     2         VWB, VEB, VSB, VNB,
     3         TWB, TEB, TSB, TNB,
     4         WWB, WEB, WSB, WNB,
     5         QWB, QEB, QSB, QNB,
     6        PPWB, PPEB, PPSB, PPNB,
     7        QCWB, QCEB, QCSB, QCNB,
     8        QRWB, QREB, QRSB, QRNB,
     9        QIWB, QIEB, QISB, QINB,
     &        QNIWB,QNIEB,QNISB,QNINB,
     1  MIX, MJX, MKX,MIXNH, MJXNH, MKXNH,KXP1NH, NSPGD,
     2  MIXM, MJXM,MKXM,MIXIC, MJXIC, MKXIC,      NSPGX,
     3  IEXMS,IICE)
       ITIMBDY(NB)=0
      BDYTIM=TIMBDY(NB)
      BDYBE = 0.0
      NMBDY = NB
c
      CALL REMOVE_NEGATIVE(QVA,QCA,QRA,QIA,QNIA,IEXMS,IICE,
     -          MIX,MJX,MKX,MIXM,MJXM,MKXM,MIXIC,MJXIC,MKXIC,
     -          IQINDX,IQCINDX,IQRINDX,IQIINDX,IQNIINDX)
      CALL XA2XB(IL,JL)
C
C
C----The cost function and forcing term related to the direct observation at the initial time.
C
C
      JDT = 0
      IF (ITSTEP(1,IDCNT(1)).EQ.0) THEN
       print*,'MM5DRY, OBS at JDT=0'
       CALL TRANSF( 1,XS,NDIMSS, UA, VA, TA, QVA,PPA, WA,
     -        MIX,MJX,MKX,
     -        QCA, QRA,MIXM, MJXM, MKXM, IEXMS,
     -        QIA,QNIA,MIXIC,MJXIC,MKXIC,IICE)
        CALL AGET(IOBS(1),1,NDIMOBSS,XSB(1))
c
        DO 90 I=1,NDIMSS
            XSB(I)=XS(I)-XSB(I)
```

```
 90     CONTINUE
c
        CALL AGET(IWGT,1,NDIMSS,XS(1))
        DO 91 I=1,NDIMSS
            VAL(1)=VAL(1)+XSB(I)*XSB(I)*XS(I)
            XSB(I)=2.0*XSB(I)*XS(I)
 91     CONTINUE
        CALL APUT0(IOUT(1),1,NDIMSS,XSB)
        IDCNT(1)=IDCNT(1)+1
        print *, 'at JDT=',JDT,'  VAL(1)=',VAL(1)
        ENDIF
C
C--- TIME INTEGRATION FOLLOWS
C
        DO 10 JDT=1,MAXSTP
C
C-----READ IN BOUNDARY CONDITIONS FOR COARSE DOMAIN (1):
C
            IF (XTIME .GE. BDYTIM) THEN
c calculate UWB(NB+1) from UWB(NB) and UWBT(NB)
c
              CALL BDY_UPDATE(NB)
              NB = NB+1
              ITIMBDY(NB)=JDT
              TBDYBE = TIMBDY(NB-1)
              BDYTIM=TIMBDY(NB)
c
              WRITE (FILENAME,'(A6,I3.3)') 'LBC__',NB
              CALL BDY_STORE( 1,FILENAME,NB,
     1             UWB,  UEB,  USB,  UNB,
     2             VWB,  VEB,  VSB,  VNB,
     3             TWB,  TEB,  TSB,  TNB,
     4             WWB,  WEB,  WSB,  WNB,
     5             QWB,  QEB,  QSB,  QNB,
     6             PPWB, PPEB, PPSB, PPNB,
     7             QCWB, QCEB, QCSB, QCNB,
     8             QRWB, QREB, QRSB, QRNB,
     9             QIWB, QIEB, QISB, QINB,
     &              QNIWB,QNIEB,QNISB,QNINB,
     1 MIX,  MJX,  MKX,MIXNH, MJXNH, MKXNH,KXP1NH, NSPGD,
     2 MIXM, MJXM,MKXM,MIXIC, MJXIC, MKXIC,      NSPGX,
     3 IEXMS,IICE)
c
              WRITE (FILENAME,'(A6,I3.3)') 'LBCT_',NB
              CALL BDY_STORE(-1,FILENAME,NB,
     1             UWBT,  UEBT,  USBT,  UNBT,
     2             VWBT,  VEBT,  VSBT,  VNBT,
     3             TWBT,  TEBT,  TSBT,  TNBT,
     4             WWBT,  WEBT,  WSBT,  WNBT,
     5             QWBT,  QEBT,  QSBT,  QNBT,
     6             PPWBT, PPEBT, PPSBT, PPNBT,
     7             QCWBT, QCEBT, QCSBT, QCNBT,
     8             QRWBT, QREBT, QRSBT, QRNBT,
     9             QIWBT, QIEBT, QISBT, QINBT,
     &              QNIWBT,QNIEBT,QNISBT,QNINBT,
     1 MIX,  MJX,  MKX,MIXNH, MJXNH, MKXNH,KXP1NH, NSPGD,
     2 MIXM, MJXM,MKXM,MIXIC, MJXIC, MKXIC,      NSPGX,
     3 IEXMS,IICE)
C
            NMBDY = NB
              PRINT *,'NB=',NB,'  BDYTIM=',BDYTIM,'  XTIME=',XTIME
            ENDIF
C write out the basic state at initial time
C
```

```
          IF (IBSTEP(IBCNT).EQ.0) THEN
            print*,'in MM5DRY, BASIC STATE at 0'
            CALL TRANS_BAS( 1,XSB,NDIMS, UA, VA, TA, QVA,PPA, WA,
     -          MIX,MJX,MKX,
     -          QCA, QRA,MIXM, MJXM, MKXM, IEXMS,
     -          QIA,QNIA,MIXIC,MJXIC,MKXIC,IICE,
     -          ITGFLG(1),TGA)
            CALL APUT0(IBASA,1,NDIMS,XSB)
            CALL TRANS_BAS( 1,XSB,NDIMS, UB, VB, TB, QVB,PPB, WB,
     -          MIX,MJX,MKX,
     -          QCB, QRB,MIXM, MJXM, MKXM, IEXMS,
     -          QIB,QNIB,MIXIC,MJXIC,MKXIC,IICE,
     -          ITGFLG(1),TGB)
            CALL APUT0(IBASB,1,NDIMS,XSB)
            IBCNT=IBCNT+1
          ENDIF
C
          IF (KTAU.EQ.0) THEN
            CALL CONADV(IDRY(1),IMOIST(1))
            CALL CONMAS(IDRY(1),IMOIST(1),MASCHK)
          ENDIF
C
C--- FOR THE INITIAL, COARSE DOMAIN ......
C
            IF (IEXEC(1).EQ.1)THEN
C
C   input: UA,VA,UEB,UEBT,...(also boundary for V,)
C
              CALL  BDYVAL(1,0,IEXEC(1))
C
C  output: UJ1,UJL,... (also for V)
C
              IEXEC(1)=2
            ENDIF
C
C--- SOLVE EQUATIONS FOR THIS DOMAIN
C
            IF(INHYD.EQ.1)THEN
C
C  input: UA,UB,......, UEB,UEBT,....
C
              CALL SOLVE3(IEXEC(1),1,0)
C
C  output: UA,UB,.....
C
            ENDIF
C
            CALL CONADV(IDRY(1),IMOIST(1))
            CALL CONMAS(IDRY(1),IMOIST(1),MASCHK)
C
C write out the basic state at time t_JDT
C
          IF (IBSTEP(IBCNT).EQ.JDT) THEN
            print*,'in MM5DRY, BASIC STATE at ',JDT
            CALL TRANS_BAS( 1,XSB,NDIMS, UA, VA, TA, QVA,PPA, WA,
     -          MIX,MJX,MKX,
     -          QCA, QRA,MIXM, MJXM, MKXM, IEXMS,
     -          QIA,QNIA,MIXIC,MJXIC,MKXIC,IICE,
     -          ITGFLG(1),TGA)
            CALL APUT0(IBASA,(IBCNT-1)*NDIMS+1,NDIMS,XSB)
            CALL TRANS_BAS( 1,XSB,NDIMS, UB, VB, TB, QVB,PPB, WB,
     -          MIX,MJX,MKX,
     -          QCB, QRB,MIXM, MJXM, MKXM, IEXMS,
     -          QIB,QNIB,MIXIC,MJXIC,MKXIC,IICE,
     -        ITGFLG(1),TGB)
            CALL APUTO(IBASB,(IBCNT-1)*NDIMS+1,NDIMS,XSB)
            IBCNT=IBCNT+1
          ENDIF
C
C write out the forcing term and calculate the cost function
C
          IF (ITSTEP(1,IDCNT(1)).EQ.JDT) THEN
            print*,'MM5DRY,OBS at JDT=',JDT
            CALL TRANSF( 1,XS,NDIMSS, UA, VA, TA, QVA,PPA, WA,
     -        MIX,MJX,MKX,
     -        QCA, QRA,MIXM, MJXM, MKXM, IEXMS,
     -        QIA,QNIA,MIXIC,MJXIC,MKXIC,IICE)
            CALL AGET(IOBS(1),(IDCNT(1)-1)*NDIMOBSS+1,NDIMOBSS,XSB(1))
c
        DO 80 I=1,NDIMSS
              XSB(I)=XS(I)-XSB(I)
 80     CONTINUE
c
            CALL AGET(IWGT,(IDCNT(1)-1)*NDIMSS+1,NDIMSS,XS(1))
        DO 81 I=1,NDIMSS
              VAL(1)=VAL(1)+XSB(I)*XSB(I)*XS(I)
              XSB(I)=2.0*XSB(I)*XS(I)
 81     CONTINUE
            CALL APUTO(IOUT(1),(IDCNT(1)-1)*NDIMSS+1,NDIMSS,XSB(1))
            IDCNT(1)=IDCNT(1)+1
            print *, 'at JDT=',JDT,' VAL(1)=',VAL(1)
          ENDIF
C
 10     CONTINUE
C
          PRINT *,NMBDY,' PERIODS BDY USED IN MM5DRY'
          print 55555,(VAL(I),I=1,NOBS_TYPE)
 55555   format(1x,'VAL=', 12(E14.5,5x))
         TVAL=0.
         DO N=1,NOBS_TYPE
         TVAL=TVAL+VAL(N)
         ENDDO
          RETURN
          END
```

5.9 The MM5 adjoint model subroutine incorporated with the forcing terms $\frac{\partial J}{\partial x(t)}$ (calculated in mm5dry.F): *adjdry.F*

```
      SUBROUTINE ADJDRY(IOUT)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C
C!!!!  THIS IS THE TANGENT LINEAR MODEL OF THE ADIABATIC VERSION of MM5  !
C        moisture variable QV, diffusion, bulk PBL  included
C
C output: initial conditions:  UA,VA,TA,QVA,PPA,WA
C      boundary conditions: FSBC,FNBC,FWBC,FEBC,FSBTC,FNBTC,FWBTC,FEBT
C                      F represents U,V,T,Q,PP,W
C  input: forecasted model state: UA,VA,TA,QVA,PPA,WA
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
#     include <parame.incl>
#     include <parmin.incl>
#     include <param2.incl>
#     include <param3.incl>
#     include <bcontrol.incl>
#     include <dusolve1.incl>
#     include <addr0.incl>
#     include <various.incl>
#     include <point3d.incl>
#     include <pointbc.incl>
#     include <nonhydb.incl>
#     include <nonhyd.incl>
#     include <minim.incl>
#     include <point2d.incl>
#     include <varia.incl>
#     include <nlsol9.incl>
#     include <addr29.incl>
#     include <uprad.incl>
C    .
C***********************************************************************
C
      DIMENSION AUA(MIX,MJX,MKX),AVA(MIX,MJX,MKX),ATA(MIX,MJX,MKX),
     -    AQCA(MIXM , MJXM,MKXM ), AQRA(MIXM , MJXM, MKXM),
     -    AQIA(MIXIC,MJXIC,MKXIC),AQNIA(MIXIC,MJXIC,MKXIC),
     -    ATGA(MIX  ,MJX  ),
     -    AQVA(MIX,MJX,MKX),APPA(MIX,MJX,MKX),AWA(MIX,MJX,KXP1NH)
      COMMON /SCRTCH/XSB(NDIMSBB),XS(NDIMS),
     2      ZZZ(NSCRTCH-NDIMSBB-NDIMS)
c   DIMENSION IDCNT(NOBS_TYPE),IOUT(NOBS_TYPE)
        DIMENSION IOUT(NOBS_TYPE)
C
        INTEGER IEXEC(MAXNES)
        CHARACTER FILENAME*9
C
C initialize
C
c duo to Kuo scheme
      DO 873 J=1,MJX
      DO 873 I=1,MIX
         RAINC(I,J)=0.
         RAINNC(I,J)=0.
 873  CONTINUE
c
        DO J=-6,6
        DO I=-6,6
          TMASK(I,J)=0.
        ENDDO
        ENDDO
c
        IDCNT(1)=NOBS
        IBCNT=NBAS-1
        IBASE_SAVE=IBCNT
        GVAL=0.
        NB=NMBDY
        IEXEC(1)=2
C
C zero "initial" condition for the adjoint model integration
C This part is not in the adjoint model checking, it should be here
C if we want to calculate the gradient
c
        CALL ZERO_VARIABLE(UA, VA, TA, QVA,PPA, WA,
     -        MIX,MJX,MKX,
     -        QCA, QRA,MIXM, MJXM, MKXM, IEXMS,
     -        QIA,QNIA,MIXIC,MJXIC,MKXIC,IICE,
     -        ITGFLG(1),TGA)
```

```
          CALL ZERO_BDY
c
          CALL ZERO_VARIABLE(UB, VB, TB, QVB,PPB, WB,
     -          MIX,MJX,MKX,
     -          QCB, QRB,MIXM, MJXM, MKXM, IEXMS,
     -          QIB,QNIB,MIXIC,MJXIC,MKXIC,IICE,
     -          ITGFLG(1),TGB)
C
C zero-out UJL,....
C
           DO 6000 K=1,MKX
           DO 6000 I=1,MIX
             UJ1 (I,K)=0.
             UJL (I,K)=0.
             UJ2 (I,K)=0.
             UJLX(I,K)=0.
      C
             VJ1 (I,K)=0.
             VJL (I,K)=0.
             VJ2 (I,K)=0.
             VJLX(I,K)=0.
 6000      CONTINUE
C
           DO 7000 K=1,MKX
           DO 7000 J=1,MJX
             UI1 (J,K)=0.
             UIL (J,K)=0.
             UI2 (J,K)=0.
             UILX(J,K)=0.
C
             VI1 (J,K)=0.
             VIL (J,K)=0.
             VI2 (J,K)=0.
             VILX(J,K)=0.
 7000      CONTINUE
c
           CALL MINIT(-1)
C
          L2D=MIX*MJX
          L3D=MIX*MJX*MKX
          L3DNH=MIXNH*MJXNH*MKXNH
          L3DNHP=MIXNH*MJXNH*KXP1NH
C
C--- TIME INTEGRATION FOLLOWS
          print*,'ITIMBDY=',ITIMBDY
C
          WRITE (FILENAME(1:9),'(A6,I3.3)') 'LBC__.',NB
          print *,' GET BDY FROM ',FILENAME
          CALL BDY_STORE(-1,FILENAME,NB,
     1  UWB9, UEB9, USB9, UNB9,
     2  VWB9, VEB9, VSB9, VNB9,
     3  TWB9, TEB9, TSB9, TNB9,
     4  WWB9, WEB9, WSB9, WNB9,
     5  QWB9, QEB9, QSB9, QNB9,
     6 PPWB9, PPEB9, PPSB9, PPNB9,
     7 QCWB9, QCEB9, QCSB9, QCNB9,
     8 QRWB9, QREB9, QRSB9, QRNB9,
     9 QIWB9, QIEB9, QISB9, QINB9,
     & QNIWB9,QNIEB9,QNISB9,QNINB9,
     1  MIX, MJX, MKX,MIXNH, MJXNH, MKXNH,KXP1NH, NSPGD,
     2  MIXM, MJXM,MKXM,MIXIC, MJXIC, MKXIC,      NSPGX,
     3  IEXMS,IICE)
c
          WRITE (FILENAME(1:9),'(A6,I3.3)') 'LBCT_.',NB
          print *,' GET BDY tendency FROM ',FILENAME
          CALL BDY_STORE(-1,FILENAME,NB,
     1    UWBT9,  UEBT9,  USBT9,  UNBT9,
     2    VWBT9,  VEBT9,  VSBT9,  VNBT9,
     3    TWBT9,  TEBT9,  TSBT9,  TNBT9,
     4    WWBT9,  WEBT9,  WSBT9,  WNBT9,
     5    QWBT9,  QEBT9,  QSBT9,  QNBT9,
     6   PPWBT9, PPEBT9, PPSBT9, PPNBT9,
     7   QCWBT9, QCEBT9, QCSBT9, QCNBT9,
     8   QRWBT9, QREBT9, QRSBT9, QRNBT9,
     9   QIWBT9, QIEBT9, QISBT9, QINBT9,
     &   QNIWBT9,QNIEBT9,QNISBT9,QNINBT9,
     1    MIX,  MJX,  MKX,MIXNH, MJXNH, MKXNH,KXP1NH, NSPGD,
     2    MIXM, MJXM,MKXM,MIXIC, MJXIC, MKXIC,       NSPGX,
     3    IEXMS,IICE)
c
          print*,'in ADJMOD, MAXSTP=',MAXSTP
          DO 10 JDT=MAXSTP,1,-1
C
          IF (IBSTEP(IBCNT).EQ.JDT-1) THEN
            print*,' get IBASA, IBCNT=',IBASA, IBCNT
            CALL AGET0(IBASA,(IBCNT-1)*NDIMS+1,NDIMS,XS)
            CALL TRANS_BAS(-1,XS,NDIMS, UA9, VA9, TA9, QVA9,PPA9, WA9,
     -           MIX,MJX,MKX,
     -           QCA9, QRA9,MIXM, MJXM, MKXM, IEXMS,
     -           QIA9,QNIA9,MIXIC,MJXIC,MKXIC,IICE,
     -           ITGFLG(1),TGA9)
            print*,' get IBASB, IBCNT=',IBASB, IBCNT
            CALL AGET0(IBASB,(IBCNT-1)*NDIMS+1,NDIMS,XS)
            CALL TRANS_BAS(-1,XS,NDIMS, UB9, VB9, TB9, QVB9,PPB9, WB9,
     -                MIX,MJX,MKX,
     -                QCB9, QRB9,MIXM, MJXM, MKXM, IEXMS,
     -                QIB9,QNIB9,MIXIC,MJXIC,MKXIC,IICE,
     -                ITGFLG(1),TGB9)
            IBCNT=IBCNT-1
            IBASE_SAVE=IBCNT
          ENDIF
C
C^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
C forcing term due to direct obs^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
C
          IF (ITSTEP(1,IDCNT(1)).EQ.JDT) THEN
            print*,'ADJMM5,Forcing  at JDT=',JDT
            CALL AGET0(IOUT(1),(IDCNT(1)-1)*NDIMSS+1,NDIMSS,XSB(1))
            CALL TRANSF(-1,XSB,NDIMSBB,AUA,AVA,ATA,AQVA,APPA,AWA,
     -            MIX,MJX,MKX,
     -            AQCA, AQRA,MIXM, MJXM, MKXM, IEXMS,
     -            AQIA,AQNIA,MIXIC,MJXIC,MKXIC,IICE)
C
            DO 4000 J=1,MJX
            DO 4000 K=1,MKX
            DO 4000 I=1,MIX
              UA (I,J,K)=AUA (I,J,K)+UA (I,J,K)
              VA (I,J,K)=AVA (I,J,K)+VA (I,J,K)
 4000       CONTINUE
            DO J=1,MJX-1
            DO K=1,MKX
            DO I=1,MIX-1
              TA (I,J,K)=ATA (I,J,K)+TA (I,J,K)
              QVA(I,J,K)=AQVA(I,J,K)+QVA(I,J,K)
            ENDDO
            ENDDO
            ENDDO
            IF (IEXMS.EQ.1) THEN
           DO J=1,MJX-1
            DO K=1,MKX
            DO I=1,MIX-1
              QCA(I,J,K)=AQCA(I,J,K)+QCA(I,J,K)
              QRA(I,J,K)=AQRA(I,J,K)+QRA(I,J,K)
            ENDDO
            ENDDO
            ENDDO
              IF (IICE.EQ.1) THEN
            DO J=1,MJX-1
            DO K=1,MKX
            DO I=1,MIX-1
                QIA(I,J,K)= AQIA(I,J,K)+ QIA(I,J,K)
               QNIA(I,J,K)=AQNIA(I,J,K)+QNIA(I,J,K)
            ENDDO
            ENDDO
            ENDDO
              ENDIF
            ENDIF
C
           DO 5000 J=1,MJXNH-1
           DO 5000 K=1,MKXNH
           DO 5000 I=1,MIXNH-1
             PPA(I,J,K)=APPA(I,J,K)+PPA(I,J,K)
 5000     CONTINUE
           DO 6002 J=1,MJXNH-1
           DO 6002 K=2,KXP1NH-1
           DO 6002 I=1,MIXNH-1
             WA(I,J,K)=AWA(I,J,K)+WA(I,J,K)
 6002     CONTINUE
           IDCNT(1)=IDCNT(1)-1
           print*,'ADJMM5,Forcing at JDT=',JDT,' end.'
          END IF
C
C ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
c added due to more than one boundary condition
c
          IF (NB.EQ.1) THEN
           TBDYBE = 0.0
          ELSE
           TBDYBE = TIMBDY(NB-1)
          ENDIF
           BDYTIM = TIMBDY(NB)
c adding ends
C
           IF (JDT.EQ.1) THEN
C                   input: UA9,VA9,UEB9,.....
             IEXEC(1)=1
             SXTIME=XTIME
             XTIME=0.
             CALL  MBDYVAL(1,0,IEXEC(1))
             IEXEC(1)=2
             XTIME=SXTIME
C                   output: UJ19,......
C
           ELSE
C  The last part in SOLVE3
C
C          HSCR1 in VARIA, PSA in POINT2D, IL,JL in VARIOUS
C
c      print*,'IL,JL,KL,ILX,JLX=',IL,JL,KL,ILX,JLX
             CALL DOTS(PSA,HSCR1,IL,JL,MIX,MJX)
C
```

```
        DO 820 K=1,KL
        DO 800 I=1,IL
          UJ19(I,K)=UA9(I,1,K)/HSCR1(I,1)
          VJ19(I,K)=VA9(I,1,K)/HSCR1(I,1)
          UJL9(I,K)=UA9(I,JL,K)/HSCR1(I,JL)
          VJL9(I,K)=VA9(I,JL,K)/HSCR1(I,JL)
          UJ29(I,K)=UA9(I,2,K)/HSCR1(I,2)
          VJ29(I,K)=VA9(I,2,K)/HSCR1(I,2)
          UJLX9(I,K)=UA9(I,JLX,K)/HSCR1(I,JLX)
          VJLX9(I,K)=VA9(I,JLX,K)/HSCR1(I,JLX)
 800    CONTINUE
        DO 810 J=1,JL
          UI19(J,K)=UA9(1,J,K)/HSCR1(1,J)
          VI19(J,K)=VA9(1,J,K)/HSCR1(1,J)
          UIL9(J,K)=UA9(IL,J,K)/HSCR1(IL,J)
          VIL9(J,K)=VA9(IL,J,K)/HSCR1(IL,J)
          UI29(J,K)=UA9(2,J,K)/HSCR1(2,J)
          VI29(J,K)=VA9(2,J,K)/HSCR1(2,J)
          UILX9(J,K)=UA9(ILX,J,K)/HSCR1(ILX,J)
          VILX9(J,K)=VA9(ILX,J,K)/HSCR1(ILX,J)
 810    CONTINUE
 820    CONTINUE
C
C           output: UJ19,.....
        ENDIF
C
C
        IF (INHYD.EQ.1)THEN
C
          CALL ASOLVE3(IEXEC(1),1,0)
C
        ENDIF
C
        IF (JDT.EQ.1) THEN
          print*,'CALL ABDYVAL'
          IEXEC(1)=1
          CALL AGET0(IBASA,1,NDIMS,XS)
          CALL TRANS_BAS(-1,XS,NDIMS, UA9, VA9, TA9, QVA9,PPA9, WA9,
     -        MIX,MJX,MKX,
     -        QCA9, QRA9,MIXM, MJXM, MKXM, IEXMS,
     -        QIA9,QNIA9,MIXIC,MJXIC,MKXIC,IICE,
     -        ITGFLG(1),TGA9)
          CALL ABDYVAL(1,0,IEXEC(1))
        ENDIF
C
        IF(NB.GT.0) THEN
        IF (JDT.EQ.ITIMBDY(NB)) THEN
          print*,'save gradient of J with respect to LBC at NB=',NB
          WRITE (FILENAME(1:9),'(A6,I3.3)') 'GLBC_',NB
          CALL BDY_STORE( 1,FILENAME,NB,
     1   UWBT,  UEBT,  USBT,  UNBT,
     2   VWBT,  VEBT,  VSBT,  VNBT,
     3   TWBT,  TEBT,  TSBT,  TNBT,
     4   WWBT,  WEBT,  WSBT,  WNBT,
     5   QWBT,  QEBT,  QSBT,  QNBT,
     6  PPWBT, PPEBT, PPSBT, PPNBT,
     7  QCWBT, QCEBT, QCSBT, QCNBT,
     8  QRWBT, QREBT, QRSBT, QRNBT,
     9  QIWBT, QIEBT, QISBT, QINBT,
     &  QNIWBT,QNIEBT,QNISBT,QNINBT,
     1   MIX,  MJX,  MKX,MIXNH, MJXNH, MKXNH,KXP1NH, NSPGD,
     2   MIXM, MJXM,MKXM,MIXIC, MJXIC, MKXIC,       NSPGX,
     3   IEXMS,IICE)
c
```

```
            CALL ABDY_UPDATE(NB)
            CALL ZERO_BDY
C
            NB=NB-1
            IF (NB.GT.0) THEN
              print *,'get BDY9 from LBC__.',NB
              WRITE (FILENAME(1:9),'(A6,I3.3)') 'LBC__.',NB
              CALL BDY_STORE(-1,FILENAME,NB,
     1    UWB9, UEB9, USB9, UNB9,
     2    VWB9, VEB9, VSB9, VNB9,
     3    TWB9, TEB9, TSB9, TNB9,
     4    WWB9, WEB9, WSB9, WNB9,
     5    QWB9, QEB9, QSB9, QNB9,
     6   PPWB9, PPEB9, PPSB9, PPNB9,
     7   QCWB9, QCEB9, QCSB9, QCNB9,
     8   QRWB9, QREB9, QRSB9, QRNB9,
     9   QIWB9, QIEB9, QISB9, QINB9,
     &   QNIWB9,QNIEB9,QNISB9,QNINB9,
     1     MIX, MJX, MKX,MIXNH, MJXNH, MKXNH,KXP1NH, NSPGD,
     2     MIXM, MJXM,MKXM,MIXIC, MJXIC, MKXIC,      NSPGX,
     3     IEXMS,IICE)
c
              WRITE (FILENAME(1:9),'(A6,I3.3)') 'LBCT_.',NB
              CALL BDY_STORE(-1,FILENAME,NB,
     1    UWBT9, UEBT9, USBT9, UNBT9,
     2    VWBT9, VEBT9, VSBT9, VNBT9,
     3    TWBT9, TEBT9, TSBT9, TNBT9,
     4    WWBT9, WEBT9, WSBT9, WNBT9,
     5    QWBT9, QEBT9, QSBT9, QNBT9,
     6   PPWBT9, PPEBT9, PPSBT9, PPNBT9,
     7   QCWBT9, QCEBT9, QCSBT9, QCNBT9,
     8   QRWBT9, QREBT9, QRSBT9, QRNBT9,
     9   QIWBT9, QIEBT9, QISBT9, QINBT9,
     &   QNIWBT9,QNIEBT9,QNISBT9,QNINBT9,
     1     MIX, MJX, MKX,MIXNH, MJXNH, MKXNH,KXP1NH, NSPGD,
     2     MIXM, MJXM,MKXM,MIXIC, MJXIC, MKXIC,      NSPGX,
     3     IEXMS,IICE)
c
             ENDIF
C
           ENDIF
           ENDIF
C
C
 10     CONTINUE
c
          CALL AXA2XB(IL,JL)
          CALL AREMOVE_NEGATIVE(QVA,QCA,QRA,QIA,QNIA,IEXMS,IICE,
     -             MIX,MJX,MKX,MIXM,MJXM,MKXM,MIXIC,MJXIC,MKXIC,
     -             IQINDX,IQCINDX,IQRINDX,IQIINDX,IQNIINDX)
          CALL AUA2UEB
          CALL AWBC
C
            print*,'save gradient of J with respect to LBC at NB=',NB
            WRITE (FILENAME(1:9),'(A6,I3.3)') 'GLBC_.',NB
            CALL BDY_STORE( 1,FILENAME,NB,
     1       UWBT, UEBT, USBT, UNBT,
     2       VWBT, VEBT, VSBT, VNBT,
     3       TWBT, TEBT, TSBT, TNBT,
     4       WWBT, WEBT, WSBT, WNBT,
     5       QWBT, QEBT, QSBT, QNBT,
     6      PPWBT, PPEBT, PPSBT, PPNBT,
     7      QCWBT, QCEBT, QCSBT, QCNBT,
     8      QRWBT, QREBT, QRSBT, QRNBT,
     9      QIWBT, QIEBT, QISBT, QINBT,
     &      QNIWBT,QNIEBT,QNISBT,QNINBT,
     1       MIX,  MJX,  MKX,MIXNH, MJXNH, MKXNH,KXP1NH, NSPGD,
     2       MIXM, MJXM,MKXM,MIXIC, MJXIC, MKXIC,       NSPGX,
     3       IEXMS,IICE)
C
C forcing term at the initial time^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
C
          IF ((ITSTEP(1,IDCNT(1)).EQ.0).AND.(IDCNT(1).GT.0)) THEN
          print*,'MM5ADJ, Forcing  at JDT=0'
            CALL AGET0(IOUT(1),(IDCNT(1)-1)*NDIMSS+1,NDIMSS,XSB(1))
            CALL TRANSF(-1,XSB,NDIMSBB,AUA,AVA,ATA,AQVA,APPA,AWA,
     -               MIX,MJX,MKX,
     -               AQCA, AQRA,MIXM, MJXM, MKXM, IEXMS,
     -               AQIA,AQNIA,MIXIC,MJXIC,MKXIC,IICE)
            DO J=1,MJX
            DO K=1,MKX
            DO I=1,MIX
              UA (I,J,K)=AUA (I,J,K)+UA (I,J,K)
              VA (I,J,K)=AVA (I,J,K)+VA (I,J,K)
            ENDDO
            ENDDO
            ENDDO
            DO J=1,MJX-1
            DO K=1,MKX
            DO I=1,MIX-1
              TA (I,J,K)=ATA (I,J,K)+TA (I,J,K)
              QVA(I,J,K)=AQVA(I,J,K)+QVA(I,J,K)
            ENDDO
            ENDDO
            ENDDO

            IF (IEXMS.EQ.1) THEN
            DO J=1,MJX-1
            DO K=1,MKX
            DO I=1,MIX-1
              QCA(I,J,K)=AQCA(I,J,K)+QCA(I,J,K)
              QRA(I,J,K)=AQRA(I,J,K)+QRA(I,J,K)
            ENDDO
            ENDDO
            ENDDO
              IF (IICE.EQ.1) THEN
            DO J=1,MJX-1
            DO K=1,MKX
            DO I=1,MIX-1
                QIA(I,J,K)= AQIA(I,J,K)+ QIA(I,J,K)
               QNIA(I,J,K)=AQNIA(I,J,K)+QNIA(I,J,K)
            ENDDO
            ENDDO
            ENDDO
              ENDIF
            ENDIF
C
            DO J=1,MJXNH-1
            DO K=1,MKXNH
            DO I=1,MIXNH-1
              PPA(I,J,K)=APPA(I,J,K)+PPA(I,J,K)
            ENDDO
            ENDDO
            ENDDO
            DO J=1,MJXNH-1
            DO K=2,KXP1NH-1
            DO I=1,MIXNH-1
              WA(I,J,K)=AWA(I,J,K)+WA(I,J,K)
            ENDDO
            ENDDO
            ENDDO
C
          END IF
c
C ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
C
C check to see whether the norm of gradient is too small
C
          CALL TRANSF( 1,XSB,NDIMSBB,UA,VA,TA,QVA,PPA,WA,
     -       MIX,MJX,MKX,
     -       QCA, QRA,MIXM, MJXM, MKXM, IEXMS,
     -       QIA,QNIA,MIXIC,MJXIC,MKXIC,IICE)
           DO 5 I=1,NDIMSBB
            GVAL=GVAL+XSB(I)*XSB(I)
 5         CONTINUE
          PRINT *,'GRADIENT = ',GVAL
          IF (GVAL .LT. 1.E-10) THEN
            PRINT *,'GRADIENT REDUCED BELOW 1.E-10'
            STOP
          END IF
c
      RETURN
      END
```

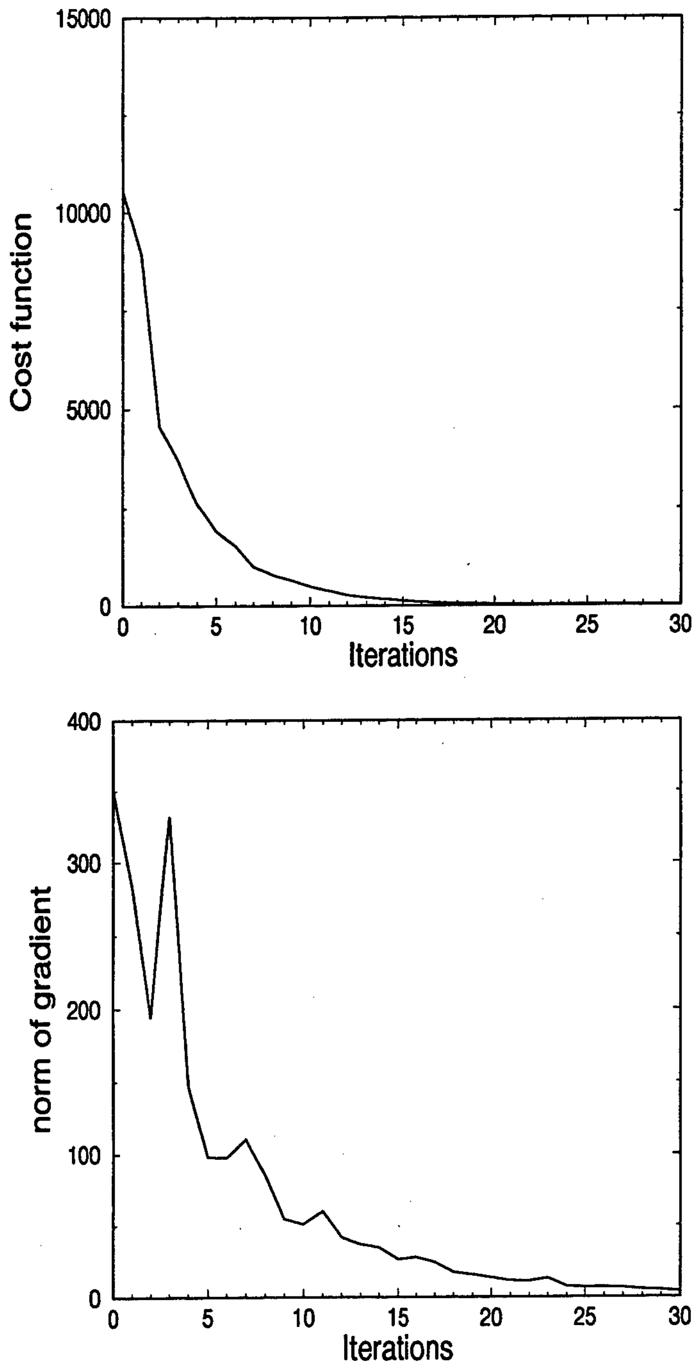


Figure 5.1 Variation of the values of the cost function (upper panel) and the norm of the gradient with respect to the ICs (lower panel) with the number of iterations.

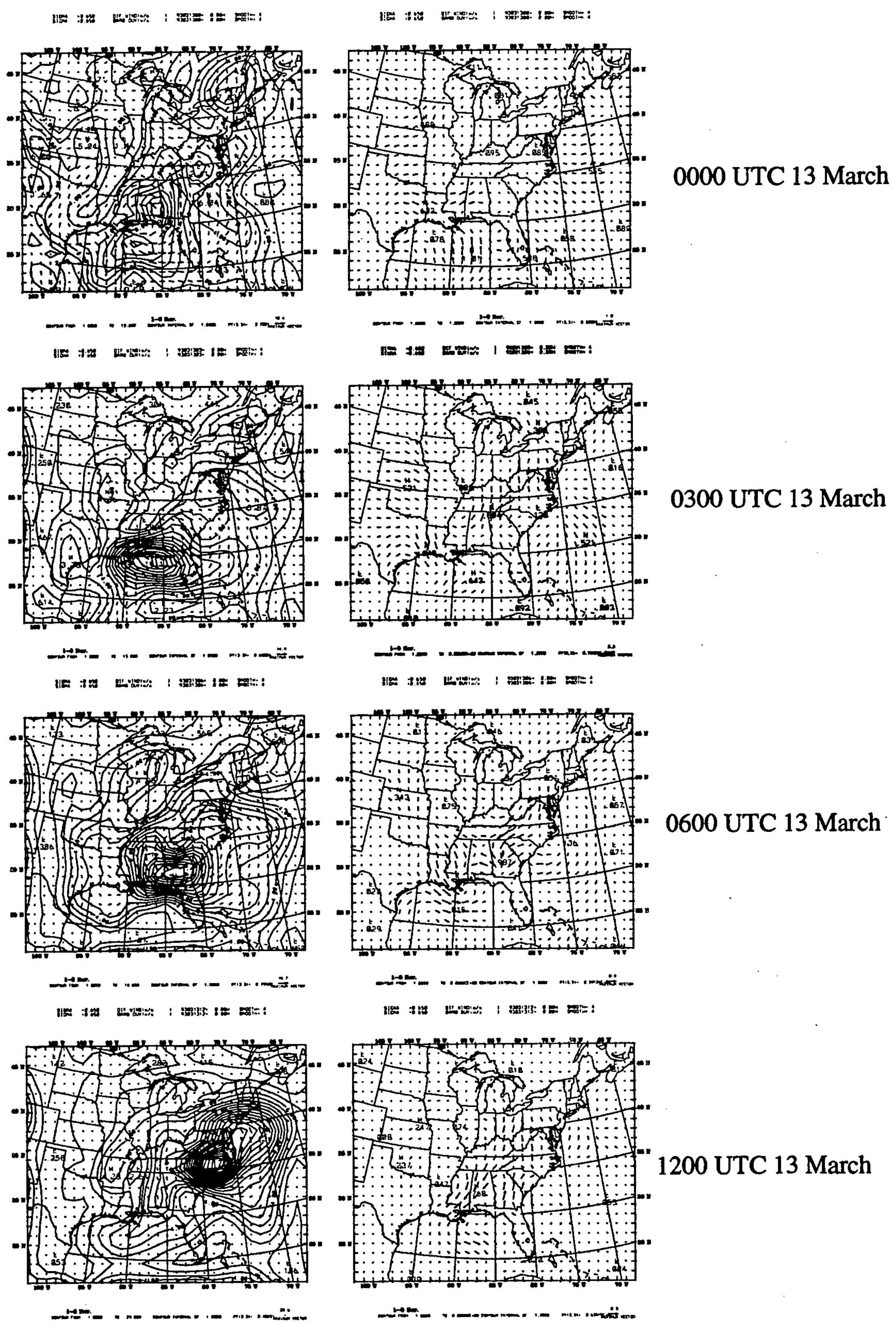


Figure 5.2: Errors in the wind fields at $\sigma = 0.65$ at the 0th (left column) and and 15th(right column) iteration from 0000 UTC 13 March to 1200 UTC 13 March 1993. Contour interval is 0.5 m/s.

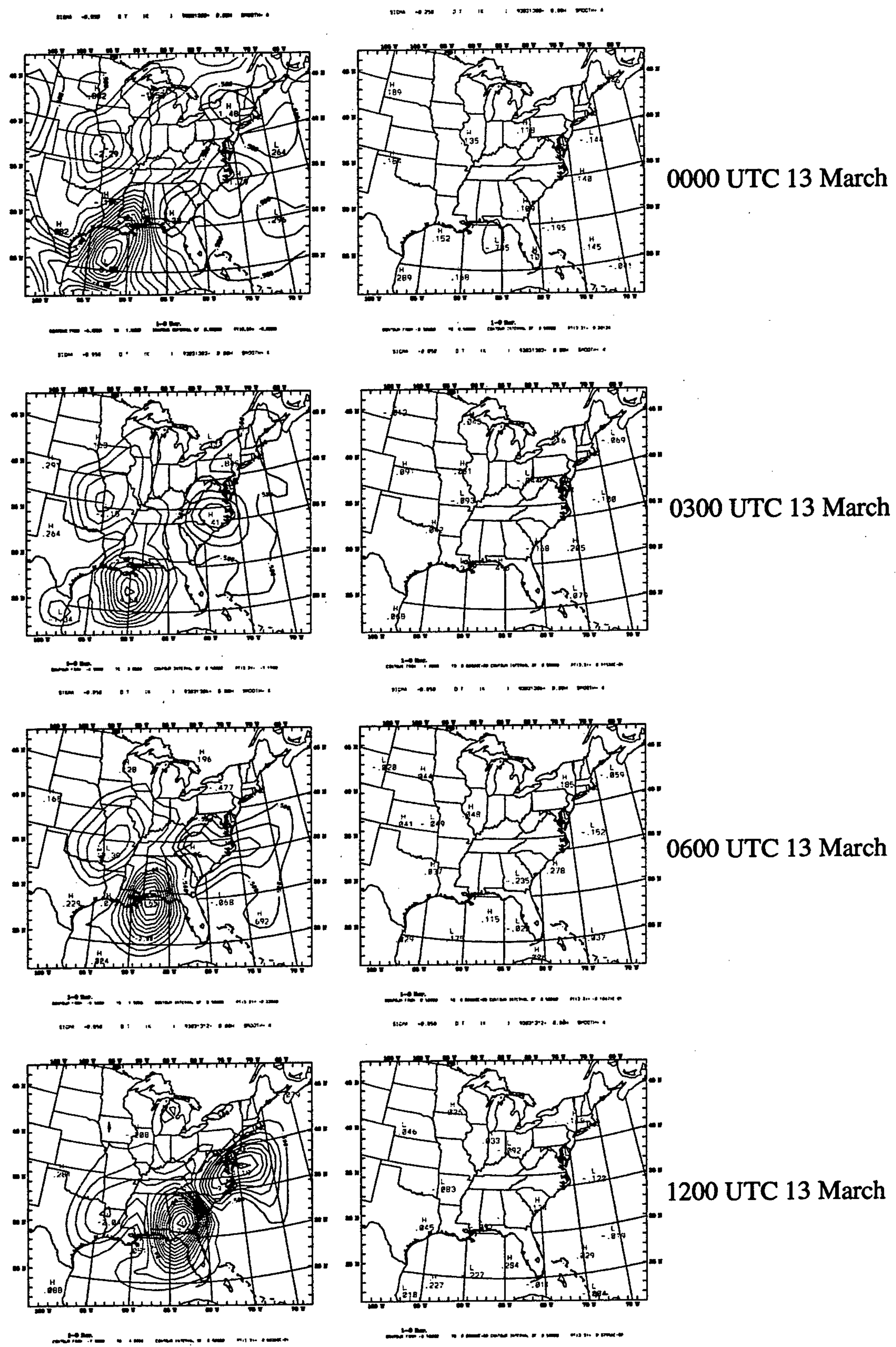


Figure 5.3: Same as Figure 5.2 except for the temperature field at σ = 0.85. Contour interval is 1.0 K.

CONCLUSION

6

6 Conclusion

The MM5 tangent linear and adjoint models can be used for a wide range of applications (see Zou et al. 1997 for a review of the most important ones). Users can make use of these tools and and adapt them to the kind of research they are interesting in. However, it is important to realize that, although *the MM5 adjoint modeling system* is a highly powerful research tool, we do not have a complete *MM5 4-dimensional variational data assimilation system* (MM5 4D-VAR system). Additional work is needed to build a complete and robust data assimilation system. Several problems still need to be solved before we can make the MM5 4D-VAR system a community mesoscale data assimilation system. These issues are numerical, physical and statistical. Following are the most important ones (that we plan to work on) in the up-coming years.

6.1 Numerical efficiency

With the present version of the code, the costs of the MM5 tangent linear and adjoint model integrations are about 2.7 and 6.9 times that of the MM5 forward model integration, respectively. The large CPU requirement in the MM5 adjoint model is caused by the J-slide structure in the MM5 model in *solve3.F* and the principle we used in the tangent linear and adjoint model development. We use minimum disk space and internal memory for saving the basic state information, which causes many repeated basic state calculations. Since at least one forward MM5 and one backward MM5 adjoint model integration is needed for every iteration in a minimization procedure, this may represent too high a computational requirement for many users who wish to use the system. It is clear that the numerical efficiency of the adjoint code that is presently released needs to be improved.There are several ways to improve the efficiency of the adjoint code: (i) change the dynamical structure of the MM5 model; (ii) multi-task and vectorize the MM5 adjoint model; (iii) improve the efficiency of the minimization procedure, and (v) adapt an incremental approach.

The computational cost of integrating the MM5 adjoint model can largely be reduced with the use of a code optimizer. A preliminary result of parallelizing the MM5 adjoint code reduced the CPU time of the adjoint model integration by half. Another way to reduce the computational cost of 4D-VAR is to improve the efficiency of the minimization procedure. This can be realized by the use of preconditioning matrices. At each iteration these matrices are applied to the gradient so that the new descent direction is more efficient than the original descent direction before transformation.This allows the minimization procedure to reach the minimum of the cost function with less iteration than a minimization without preconditioning. The best preconditioner is the inverse of the Hessian matrix, which is, however, not accessible and can certainly not be fully inversed due to its large dimension for a primitive equation model like MM5. Preconditioning matrices, therefore, must be estimated according to the problem that has to be solved.

The efficiency improvement that will result from vectorization, multitasking and preconditioning will not, however, be enough to allow any kind of high resolution experiments such as cloud-scale Doppler radar data assimilation. For that type of research, the assimilation algorithm should be modified and some approximations would inevitably be necessary. One very interesting approach is the so-called incremental approach. In this approach the nonlinear model is only used to define the basic-state trajectory while the tangent linear model is used in conjunction with the adjoint in the minimization loop. Both the tangent linear and adjoint models are used either with a low resolution or with degraded physics, requiring much less CPU time than models with full resolution and physics. This gain in computational cost is amplified by the iterative nature of the minimization process, which only uses the low-cost degraded tangent linear
and adjoint models. Some operational centers have estimated that one order of magnitude reduction of the computational cost can be expected by using such an approach when combined with a good preconditioner. From a technical point of view, the incremental approach can be easily implemented. It suffices to replace the nonlinear model with its tangent linear approximation in the minimization routines.

6.2 Observation operators

In the standard code which is being released at the first MM5 adjoint tutorial, only direct observations (model variables) are assimilated into the MM5 model. However many indirect observations (diagnostic quantities which are nonlinear functions of model variables) such as satellite radiances, surface rainfall amount, precipitable water, and radar reflectivity data are, or will be, available on a routine basis for mesoscale studies. In order to make use of these sources of observational information, various observational operators need to be developed along with their adjoint counterparts. This means that for each kind of data, a numerical operator that represents these observations from the MM5 model variables must be developed and its tangent linear and adjoint operators coded. The MM5 Adjoint Modeling System Group is currently working on the assimilation of five different kinds of indirect observations:

1) GPS/MET radio occultations,
2) GPS precipitable water profiles,
3) Doppler radar observations,
4) Surface rainfall observations, and
5) Satellite radiances

It is hoped that this research will lead to the development of a series of observation operators and enable us to incorporate more observations into the MM5 model. This new software, and that developed by outside users, will be added to the present version of the MM5 adjoint modeling system as soon as it becomes available and well-tested.

6.3 Adjoint physics

The current version (released in the summer of 1998) of the MM5 adjoint modeling system includes model dynamics, horizontal and vertical diffusions, a bulk PBL process, dry-convective adjustment, grid-resolvable precipitation, the Kuo cumulus parameterization, surface friction, a time-splitting scheme, and surface fluxes.

The tangent linear and adjoint operators of the following physical processes have already been developed, tested, and used for a few case studies: the adjoints of the Grell cumulus parameterization; a microphysics scheme (Dudhia's scheme), and a high-resolution PBL scheme (the Blackadar scheme). These physical processes need further testing before they can be released for more general use.

6.4 Background information

The current MM5 adjoint modeling system does not have the necessary error statistics. We realize that the background field and its error covariance matrix is crucial information to ensure successful data assimilation, especially when sufficient observations are not available.The background term in the cost function for mesoscale data assimilation needs to be developed, and its importance needs to be assessed. With the desire to apply the MM5 adjoint modeling system over a wide range of horizontal grid sizes and scales and over a wide range of geographical locations, as with the MM5 model, the development of a proper background term for the MM5 assimilation 4D-VAR system is a highly challenging task. This background term is also the basis for building a 3D-VAR system which may be preferably used in operational or real-time data assimilation experiments due to its low computational cost. The 3D-VAR system may replace some of the preprocessing in the MM5 modeling system. Such an effort is now being undertaken by MPG/MMM/NCAR.

The development of the MM5 4D-VAR system was a tremendous effort of many person-years. Most of the development effort was supported by external fundings. The MM5 4D-VAR system as it stands now is a highly useful research tool for data assimilation research, sensitivity analysis, and dynamic studies. However, much work remains to be done in making the MM5 4D-VAR system a more complete data assimilation system, and in making it a versatile tool for mesoscale research and semi-operational application. With this initial release of the MM5 4D-VAR system to the community, we are encouraging user input and modification to begin at once. Future development and improvement of the system will depend heavily, not only on the personnel at NCAR, but also on the user community in keeping the MM5 4D-VAR system at the cutting edge of mesoscale data assimilation area.

References

Anthes, R. A. and T. T. Warner, 1978: Development of hydrodynamic models suitable for air pollution and other mesometeorological studies. *Mon. Wea. Rev.*, **106**, 1045-1078.

Dudhia, J., 1993: A nonhydrostatic version of the Penn State-NCAR mesoscale model: Validation tests and simulation of an Atlantic cyclone and cold front. *Mon. Wea. Rev.*, **121**, 1493-1513.

Grell, G. A., J. Dudhia and D. R. Stauffer, 1993: A description of the fifth-generation Penn State/NCAR mesoscale model (MM5). *NCAR Technical Note*, NCAR/TN-398+STR, 117 pp.

Zou, X., and Y.-H. Kuo, 1996: Rainfall assimilation through an optimal control of initial and boundary conditions in a limited-area mesoscale model. *Mon. Wea. Rev.*, **124**, 2859-2882.

Zou, X., Navon, I. M., Berger, M., Phua, Paul K. H., Schlick, T., and Le Dimet, F. X. 1993. Numerical experience with limited-memory quasi-Newton andtruncated Newton methods. *SIAM J. on Optimization*, **3**, 582-608.

Zou, X., F. Vandenberghe, M. Pondeca and Y.-H. Kuo, 1997: Introduction to adjoint techniques and the MM5 adjoint modeling system. *NCAR Technical Note*, NCAR/TN-435+STR, 117 pp.